Citrix Virtual Apps and Desktops 2311 SDK

Contents

Citrix Virtual Apps and Desktops 2311 SDK

March 11, 2024

> **Note:**
>
> Citrix Virtual Apps and Desktops was formerly XenApp and XenDesktop.
>
> The new product and component names stem from the expanding Citrix portfolio and cloud strategy.

Implementing this transition in our products and their documentation is an ongoing process. Your patience during this transition is appreciated. For more detail about our new names, see https://www.citrix.com/about/citrix-product-guide/.

Citrix Virtual Apps and Desktops provide an SDK based on a number of Microsoft Windows PowerShell cmdlets that allows you to perform the same tasks as you would with the Citrix Studio console, together with tasks you cannot do with Studio alone.

Differences in policy rules

There are differences between the SDK and the Studio console in terms of policy rules. Entitlement and assignment policy rules are independent entities in the SDK; in the console, these entities are not visible as they are seamlessly merged with the Delivery Group. Also, access policy rules are less restrictive in the SDK.

Use the SDK

The SDK comprises of a number of PowerShell cmdlets packaged both as modules and snap-ins. These are installed automatically by the installation wizard when you install the Controller or Studio components.

To access and run the cmdlets:

1. Start a shell in Windows PowerShell.

   To start a shell from the console, click **Studio**, select the PowerShell tab, and click on **Launch PowerShell**.

   You must run the shell or script using an identity that has Citrix administration rights. Although members of the local administrators group on the Controller automatically have full administrative privileges to allow Citrix Virtual Apps and Desktops to be installed, Citrix recommends

that for normal operation, you create Citrix administrators with the appropriate rights, rather than use the local administrators account.

2. To use SDK cmdlets within scripts, set the execution policy in PowerShell. For more information about PowerShell execution policy, see your Microsoft documentation.
3. To use this SDK through the snap-ins, add the snap-ins you require into the PowerShell environment using the **Add-PSSnapin** command in the Windows PowerShell console. **This step is unnecessary if you want to use the cmdlets in this SDK through the modules**.

   V1 and V2 denote the version of the snap-in (Citrix Virtual Apps and Desktops snap-ins are version 2.).

   For example, type:

   ```
   Add-PSSnapin Citrix.ADIdentity.Admin.V2
   ```

   To import all the cmdlets, type:

   ```
   Add-PSSnapin Citrix.*.Admin.V*
   ```

   After importing, you have access to the cmdlets and their associated help.

   For an example of a typical use case, see Get started with the SDK.

Documentation notes

- In this API documentation, some property types are marked with a question mark (?). This notation indicates that those data types are nullable.

about_CitrixCommonParameters

March 11, 2024

Short description

Describes the parameters that can be used with most Citrix DaaS cmdlets.

Long description

You can use the common Citrix parameters with most Citrix DaaS cmdlet, but they might not have an effect on all cmdlets.

Many of these parameters are used internally for passing trace information between processes.

The following list displays the common Citrix parameters.

- **AdminAddress**
- **AdminClientIP**
- **BearerToken**
- **TraceParent**
- **TraceState**
- **VirtualSiteId**

Common Citrix parameter descriptions

-AdminAddress

Specifies the address of a XenDesktop controller the PowerShell snap-in will connect to. You can provide this as a host name or an IP address.

Type	String
Position	Named
Default value	Localhost. Once a value is provided by any cmdlet, this value becomes the default.
Required	False
Accept pipeline input	False
Accept wildcard characters	False

-AdminClientIP

Specifies the Client IP of the calling user.

Type	String
Position	Named
Default value	None
Required	False
Accept pipeline input	False
Accept wildcard characters	True

-BearerToken

Specifies the bearer token assigned to the calling user.

Type	String
Position	Named
Default value	None
Required	False
Accept pipeline input	False
Accept wildcard characters	True

-TraceParent

Specifies the OpenTelemetry distributed tracing trace parent for this cmdlet call.

This property should follow the W3C Trace Context format for a traceparent header.

Note this property will also detect any ambient Activity provided by the caller provided by Activity.Current.

Type	String
Position	Named
Default value	None
Required	False
Accept pipeline input	False
Accept wildcard characters	True

-TraceState

Specifies the OpenTelemetry distributed tracing trace state for this cmdlet call.

This property will be ignored if no TraceParent is specified.

This property should follow the W3C Trace Context format for a tracestate header.

Type	String
Position	Named

Default value	None
Required	False
Accept pipeline input	False
Accept wildcard characters	True

-VirtualSiteId

Specifies the virtual site the PowerShell snap-in will connect to.

Type	String
Position	Named
Default value	None
Required	False
Accept pipeline input	False
Accept wildcard characters	True

Citrix Group Policy Provider

March 11, 2024

The Citrix Group Policy Provider (referred as the 'provider'in the rest of the document) is a Windows PowerShell snap-in that is implemented as a PowerShell provider. For more information on providers, see Microsoft documentation.

Citrix provider supports common provider commands such as the `Get-ChildItem` command and the `Set-ItemProperty` command.

Prerequisites

- Familiarity with Windows providers

Install the Citrix Group Policy Provider

- **For on-premises installations** - The provider is installed as part of the Citrix controller installation when installing Citrix Studio. You can install Citrix Studio on a supported Windows machine.
- **For Citrix Cloud deployments** - The Citrix Remote PowerShell SDK is required. For more information about this SDK, see SDKs and APIs documentation.

Manage policies using the provider

To manage policies using the provider, you must be a Citrix administrator with one of the following privileges:

- Full administrators with default rights to manage policies.
- Custom administrators with the permission to manage and view policies.

To begin managing policies using the provider, you must first load the provider to PowerShell session, which might be a Windows PowerShell console or a script to use the provider. After loading the provider to PowerShell session, you must mount the provider to a drive.

Step 1: Load the provider

To load the provider, navigate to the PowerShell console and run the following command:

```
1     Import-Module Citrix.GroupPolicy.Commands
```

Alternatively, being a PowerShell snap-in, you can also load the provider using the following Add-PsSnapin command:

```
1     Add-PsSnapin Citrix.Common.GroupPolicy
```

> **Note:**
>
> If the provider does not load using the `Import-Module Citrix.GroupPolicy.Commands` or the `Add-PsSnapin Citrix.Common.GroupPolicy` command you might not have the provider installed in your system..

Step 2: Mount the provider

PowerShell providers provide a user interface similar to a file system. At the top of the hierarchy is a drive, which is similar to a file system drive. After the provider is loaded, use commands to mount a provider to a drive so that its objects can be accessed. The four parameters of the command are the following:

- The `-PsProvider` parameter specifies the name of the provider. The value must be specified and must be `CitrixGroupPolicy`.
- The `-Name` parameter specifies the name of the drive.
- The `-Root` parameter specifies the root of the drive in the hierarchy of the provider's object tree. Usually the root of the tree is specified.
- The `-Controller` parameter specifies the Citrix controller to use.

Mount the provider to a drive using the following command:

- On-premises: Use `localhost` to connect to a controller running on the same machine. When a remote host is specified, the remote host must be a Citrix controller and the necessary ports must be open for remotely connecting to the controller. The signed in user must be also an administrator of the remote site. If a remote connection is to be used, the fully qualified domain name (FQDN) of the remote controller with an optional port number must be specified. If no port number is specified, by default port 80 is used. To mount a drive for on-premises deployments, run the following command:

```
New-PsDrive -PsProvider CitrixGroupPolicy -Name GP -Root \ -
   Controller localhost
```

- Cloud: If the controller is a cloud controller, the connection must be made using the Citrix Remote PowerShell SDK. Run the following command to connect to a cloud environment. Before running this command, ensure that you are authenticated using the `Get-XDAuthentication` command or you have set an authentication profile. For example, `Set-XdCredentials`.

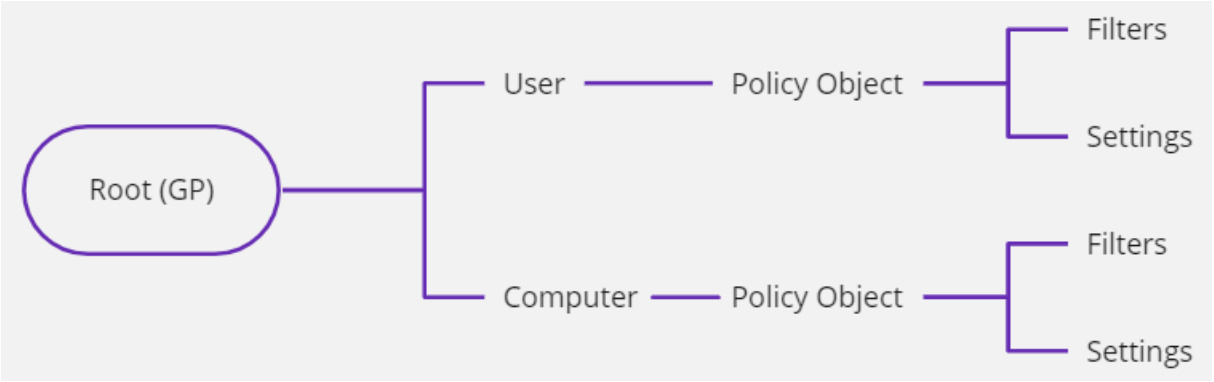
```
New-PsDrive -PsProvider CitrixGroupPolicy -Name GP -Root \ -
   Controller $GLOBAL:XDSDKProxy -BearerToken $GLOBAL:XDAuthToken
```

After the **New-PsDrive** command is successfully run, the complete policy information is created as a file system under the name provided.

```
cd GP:
```

Navigating the object tree

Objects in the provider form a tree. At the top, the root of the tree is the drive root. In the root, there are two nodes named as **Computer** and **User** nodes respectively. The computer policies are stored in the **Computer** node. The user policies are stored in the **User** node.



The definition and difference between computer policies and user policies are listed in the following table:

Computer policy	User policy
Computer policies are policy objects that include only computer settings. Computer settings are settings that are validated for all user connections in the VDAs. Computer settings are applied only at the time of VDA reboots, restart of the Citrix Broker Agent (Citrix Desktop Service), or at 90 minute intervals.	User policies are policy objects that include only user settings. User settings are settings that are validated at every user sign-in or reconnect.

Policy objects are available either in the computer or user node. There is **Settings** and **Filters** available for each policy object.

Policy filters are objects that determine how a policy is applied. The policy engine running on a VDA does the filtering calculation using a set of data called filter evidence. Filter evidence includes data such as the user name and the client name.

The objects in the **Settings** node of a policy node are settings and are organized using categories. Each category node includes subcategories or individual setting objects, which are the leaves of the tree under the **Settings** node. Nodes for all the settings are always defined and they cannot be created or removed. You can only configure a **Settings** node by changing the node's properties.

The objects under the **Filters** node of a policy node are filters, that you can create, modify, and delete.

Create, modify, or delete a policy

Create a policy

To create a computer policy, under the **Computer** node use the **New-Item command**. For example, the following command creates a policy named `Policy123`:

```
PS GP:\Computer\> New-Item Policy123

Name           : Policy123
Description    :
Enabled        : False
Priority       : 7
MergedPriority : 0
PSPath         : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\
    Computer\Policy123
PSParentPath   : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\
    Computer
PSChildName    : Policy123
PSDrive        : GP
PSProvider     : Citrix.Common.GroupPolicy\CitrixGroupPolicy
PSIsContainer  : True
```

After the policy is created, the properties of the policy object are displayed.

Modify a policy

You can only modify the `Description`, `Enabled`, and the `Priority` properties for a policy. Do not set the `MergedPriority` property because only the Citrix Studio uses this property.

To modify a property of a policy, use the `Set-ItemProperty` command. For example, the following command sets the description of the policy to `test policy`:

```
PS GP:\Computer\> Set-ItemProperty -Path Policy123 -Name Description -
    Value "test policy"
PS GP:\Computer\> Get-ItemProperty Policy123

Name           : Policy123
Description    : test policy
Enabled        : False
Priority       : 7
MergedPriority : 0
PSPath         : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\
    Computer\Policy123
PSParentPath   : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\
    Computer
PSChildName    : Policy123
PSDrive        : GP
PSProvider     : Citrix.Common.GroupPolicy\CitrixGroupPolicy
```


To display the result of the `Set-ItemProperty` command, use the `Get-ItemProperty` command.

Delete a policy

To delete an existing policy, use the `Remove-Item` command. The following command removes the policy named `Policy123`:

```
PS GP:\Computer\> Remove-Item Policy123

Confirm
The item at GP:\Computer\Policy123 has children and the Recurse
    parameter was not specified. If you continue, all
children will be removed with the item. Are you sure you want to
    continue?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (
    default is "Y"): a
PS GP:\Computer\>
```

The policy objects might have child objects. If you try to delete a policy object that has child objects, the child objects are also removed. As a result, the PowerShell needs a confirmation for the removal of all child objects.

To avoid the prompt, you can use the following `-Recurse` parameter:

```
PS GP:\Computer\> Remove-Item Policy123 -Recurse
PS GP:\
Computer\>
```

Select setting for a policy

After a new policy is created, setting objects are automatically created in the new policy. The following example shows the top level settings for a newly created policy:

```
PS GP:\User\> New-Item Policy123

Name           : Policy123
Description    :
Enabled        : False
Priority       : 102
MergedPriority : 0
PSPath         : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123
PSParentPath   : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User
PSChildName    : Policy123
PSDrive        : GP
PSProvider     : Citrix.Common.GroupPolicy\CitrixGroupPolicy
PSIsContainer  : True
```



```
14
15 PS GP:\User\> cd Policy123
16
17 PS GP:\User\Policy123> dir
18 Filters
19 Settings
20 PS GP:\User\Policy123> cd .\Settings\
21
22 PS GP:\User\Policy123\Settings\> dir
23 ICA
24 Receiver
25 UserProfileManager
26 VirtualDesktopAgent
27 PS GP:\User\Policy123\Settings\>
```

There are no individual settings in the top level objects under the **Settings** node. All the objects are container objects created from the categories of the settings. To find the setting that you want to select for your policy, browse the folders, or use the settings reference. For more information about where to find the settings reference and how to use it, see the Setting definitions reference section.

For example, if you want to enable the setting **Readonly clipboard**, you can use the following `Set-ItemProperty` command:

```
1 PS GP:\User\Policy123\Settings\> Set-ItemProperty -Path .\ICA\
     ReadonlyClipboard\ -Name State -Value Enabled
2 PS GP:\User\Policy123\Settings\> Get-ItemProperty -Path .\ICA\
     ReadonlyClipboard\
3 State        : Enabled
4 PSPath       : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
     Policy123\Settings\ICA\ReadonlyClipboard
5 PSParentPath : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
     Policy123\Settings\ICA
6 PSChildName  : ReadonlyClipboard
7 PSDrive      : GP
8 PSProvider   : Citrix.Common.GroupPolicy\CitrixGroupPolicy
```

To display the result of the `Set-ItemProperty` command, use the `Get-ItemProperty` command.

As shown in the example, to select a setting, set the value of the `State` property of the setting to `Enabled`. For settings whose value type is bool, for example, `ReadonlyClipboard`, only the `State` property needs to be set. For settings whose value type is not bool, the `Value` property might need to be set. If the `Value` property is not set, the default value of the setting is used.

> **Note:**
>
> You can only use the internal name of the setting. For many settings, the internal names and the displayed English names are different. To find the correct internal names, use the setting definitions reference section.

PowerShell auto-completion can be used in navigating the settings tree, which saves much typing. Some settings are under several layers of categories.

Setting objects can only be selected by setting the value of the `State` property to `Enabled` or `Allowed`. A setting can only be deselected by setting the value of `State` to `Disabled` or `Prohibited`. Set the value of `State` to `UseDefault` to reset the setting. Setting objects cannot be created or removed.

Deselect a setting for a policy

To deselect a setting, set the `State` property to `Disabled`, `Prohibited`, or `UseDefault` as described in the following command:

```
PS GP:\User\Policy123\Settings\> Set-ItemProperty -Path .\ICA\
    ReadonlyClipboard\ -Name State -Value Disabled
PS GP:\User\Policy123\Settings\> Get-ItemProperty -Path .\ICA\
    ReadonlyClipboard\
State        : Disabled
PSPath       : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Settings\ICA\ReadonlyClipboard
PSParentPath : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Settings\ICA
PSChildName  : ReadonlyClipboard
PSDrive      : GP
PSProvider   : Citrix.Common.GroupPolicy\CitrixGroupPolicy
```

Create, modify, or delete a filter for a policy

Create a filter for a policy

Filter objects are present in the **Filters** node of a policy. Unlike settings, filters are not prepopulated, except the top level nodes for each filter type. Create a filter object and specify its properties to create a filter for a policy. The following example defines a client name filter:

```
PS GP:\User\Policy123\Filters\ClientName\> New-Item ClientNameFilter1
Synopsis      : Allow - ClientNameFilter1
FilterValue   : ClientNameFilter1
Name          : ClientNameFilter1
FilterType    : ClientName
Mode          : Allow
Enabled       : True
Comment       :
PSPath        : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Filters\ClientName\ClientNameFilter1
PSParentPath  : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Filters\ClientName
PSChildName   : ClientNameFilter1
```



```
PSDrive       : GP
PSProvider    : Citrix.Common.GroupPolicy\CitrixGroupPolicy
PSIsContainer : False
```

Modify a filter for a policy

You can only modify the `FilterValue`, `Name`, `Mode`, and `Enabled` properties. To modify a filter property, use the `Set-ItemProperty` command. The following command sets the value of the client name filter to `Windows11`:

```
PS GP:\User\Policy123\Filters\ClientName\> Set-ItemProperty
    ClientNameFilter1 -Name FilterValue -Value "Windows11"
PS GP:\User\Policy123\Filters\ClientName\> Get-ItemProperty
    ClientNameFilter1
Synopsis     : Allow - Windows11
FilterValue  : Windows11
Name         : ClientNameFilter1
FilterType   : ClientName
Mode         : Allow
Enabled      : True
Comment      :
PSPath       : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Filters\ClientName\ClientNameFilter1
PSParentPath : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123\Filters\ClientName
PSChildName  : ClientNameFilter1
PSDrive      : GP
PSProvider   : Citrix.Common.GroupPolicy\CitrixGroupPolicy
```

Delete a filter for a policy

To remove a filter, use the `Remove-Item` command. No child objects are present under a filter object, which is a leaf node. As a result, no need to use the `-Recurse` switch:

```
PS GP:\User\Policy123\Filters\ClientName\> Remove-Item
    ClientNameFilter1
PS GP:\User\Policy123\Filters\ClientName\>
```

Setting policy priorities

One of the properties of the policy object is Priority, which is an integer. The priority of a policy specifies the precedence in calculating the result of applying all the policies in the site. The priority number is automatically assigned when a new policy is created. The larger the priority number, the lower the priority of the policy.

When a new policy is created, the priority of the policy is assigned to the lowest. This setting means that the priority number of the policy is assigned to one larger than the existing lowest priority number. Priority numbers are 1-based and there are no gaps allowed. This setting means that the largest priority number always is the number of policies. For example, if there are three policies defined, the priority of the next policy will be 4.

Modify policy priorities

To update the priority of a policy, you use the `Set-ItemProperty` command that sets the priority of a policy. For example:

```
PS GP:\User> Get-ItemProperty Policy123
Name           : Policy123
Description    :
Enabled        : False
Priority       : 102
MergedPriority : 0
PSPath         : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123
PSParentPath   : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User
PSChildName    : Policy123
PSDrive        : GP
PSProvider     : Citrix.Common.GroupPolicy\CitrixGroupPolicy

PS GP:\User> Set-ItemProperty Policy123 -Name Priority -Value 10
PS GP:\User> Get-ItemProperty Policy123

Name           : Policy123
Description    :
Enabled        : False
Priority       : 10
MergedPriority : 0
PSPath         : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User\
    Policy123
PSParentPath   : Citrix.Common.GroupPolicy\CitrixGroupPolicy::GP:\User

PSChildName    : Policy123
PSDrive        : GP
PSProvider     : Citrix.Common.GroupPolicy\CitrixGroupPolicy
```

In this example, the newly created `Policy123` is automatically assigned the priority number 102. To move it to priority 10, set the value of `Priority` to 10.

Assigning the priority of a policy changes the priorities of all the policies behind it. In this example, the policy that was at priority 10 is now at priority 11. The policy that had priority number 11 is now at priority number 12, and so on. The priorities of the policies ahead of the new priority value are not changed. This setting means that the priorities of the policies at 1 through 9 are not changed.

In this provider, policy priorities can only be changed one at a time.

Removing a policy results in the policies with lower priorities to get moved up by one.

Priority number cannot be more than the total number of policies. For example, if we have 102 policies, no policy can be set to priority 103 or larger. If the priority of a policy is assigned a number greater than the number of policies available, the priority of the policy is set to the number of existing policies. This behavior is convenient when you want to move a policy to the lowest priority. For example, if there are only 10 policies and if you set the priority of the `Unfiltered` policy to 10000, the priority of the `Unfiltered` policy will be set to 10, which is the lowest priority.

Merge user and computer policies

The provider has the user and computer policies organized separately. But in the policy management consoles (Citrix Studio), there is no distinction between these two types of policies. A computer and a user policy that have the same name are merged in the Citrix Studio as one policy.

The merge is done using the name only. If a computer or user policy does not have a counterpart in the other policy type, it exists as a policy in Citrix Studio, although it has only computer or user settings configured in it.

Merging the user and computer policies of the same name imposes the following requirements to create and edit policies in the provider:

- No priority inversion is allowed. Priority inversion is a condition where the computer and user portions of two policies are in reverse orders. For example, consider we have the user policies `Policy0` and `Policy1` with priority numbers 1 and 2 and the computer policies `Policy0` and `Policy1` with priority numbers 6 and 3. In this example, the priority inversion occurs between `Policy0`and `Policy1`. The policy numbers do not have to be consecutive and they do not have to be the same numbers. If the computer policies `Policy0` and `Policy1` have priorities 3 and 6, there is no priority inversion occurs between `Policy0` and `Policy1`. For the entire set of policies, there must be no priority inversion between any two policies.
- Filters must be consistent. The computer and user portions of the same filter type must have filters that are consistent. Consider a scenario where the user policy `Policy0` has a filter to allow delivery group `Dg0` and the computer policy `Policy0` has a delivery group filter that denies `Dg0`. In this example, there is filter inconsistency between the two portions of `Policy0`.
- Policies must be of the same state. Both the user and computer value of a policy must be set to the same state. They must be both either enabled or disabled.

The Citrix Studio detects these conditions and refuses to display the policies when such a condition exists.

> **Note:**
>
> The `MergedPriority` property is used by Citrix Studio to maintain the priority numbers of the

> merged policies. The provider must not alter the `MergedPriority` property. It is reset every time policies are opened in the Citrix Studio.

Setting definitions reference

Policies are the containers of settings. VDAs validates settings as the configurable values to achieve the desirable effects. In each VDA release, with newly added VDA features, there are new settings that allow Citrix administrators to configure these features.

There are already over 400 settings. Although in the UI, settings can be searched, it's not easy to find the setting that you want to configure in the provider. When you write a script, you might use the Citrix Studio to find the settings. But when you do not have access to the Citrix Studio, you need a different source to navigate the setting hierarchy.

You can find a complete reference of Citrix policy settings in the Policy documentation.

Recommendation on efficient use of the provider

Behind the scenes, the provider automatically saves every change whenever a PowerShell command like `New-Item` or `Set-ItemProperty` is run. The entire policy data is stored as a binary blob. As a result, every change might result in thousands of bytes written to the controller's site database.

If you connect to the local computer or if you have few number of policies, settings, and filters configured, you might not notice any delays. But if you edit a large blob that has dozens of policies and many settings and filters, or if the controller is remote, you might notice delays after a change is made.

The unnecessary saves to the database can be turned off. Set the provider's drive property `AutoWriteBack` value to `$false` to clear the unnecessary saves:

```
PS GP:\User> (Get-PSDrive GP).AutoWriteBack
True
PS GP:\User> (Get-PSDrive GP).AutoWriteBack = $false
PS GP:\User> (Get-PSDrive GP).AutoWriteBack
False
PS GP:\User>
```

After the `AutoWriteBack` is set to `$false`, you must save the changes because changes will not be automatically saved. You can save the changes using the drive's `Save()` method:

```
PS GP:\User> (Get-PSDrive GP).Save()
PS GP:\User>
```

> **Note:**
>
> Turning off auto save is recommended in scripts. It can significantly improve the execution speed

> of the scripts.

FAQ

How do filters get applied?

This question is not just limited to using the provider.

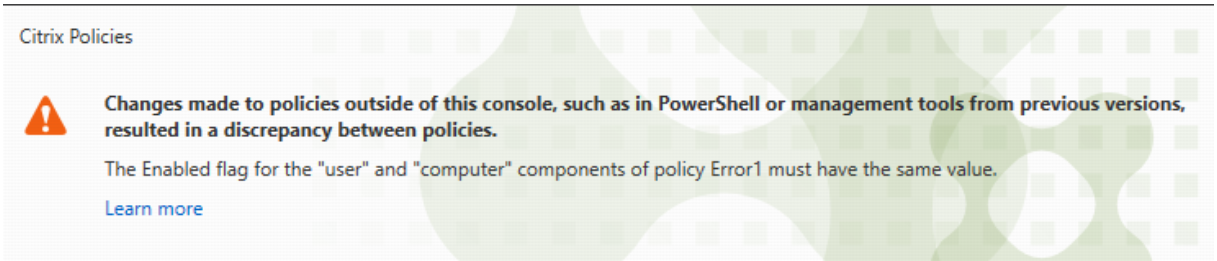
The mode of a filter specifies how a filter must be applied. The value `Allow` means that the policy will be applied if the filter condition is met. The value `Deny` means that the policy will not be applied when the filter condition is met. Among multiple filters of the same type, `Deny` take precedence over `Allow`. For example, consider a scenario when the user `user1` is a member of the users group and a user filter allows user group `users` and another filter denies the user `user1`. In this scenario, when the filters are evaluated, the net result on evaluating the user filters is to not apply the policy if the user is `user1`.

Among filters of the same type, the filters are ANDed together. For example, among all the user filters, the net result of evaluating the filters is when all the filter conditions are met, with the deny filters having precedence over allow filters.

Among filters of different types, the filters are ORed together. For example, consider a scenario where you have some delivery group filters and some user filters and the net result of evaluating the user filters is denying `user1` and allowing the delivery group `dg1`. In this scenario, the policy is applied if the VDA is in `dg1` and only its user policy portions are not applied if `user1` connects to the VDA.

Why errors seen in Citrix Studio after changes are made in provider?

After the policies are updated using the provider, an error appears in the Citrix Studio similar to the following one:



As the error indicated, there is a policy named `Error1` but actually there are two separate policies named `Error1` in the provider. The user policy `Error1` might have the `Enabled` property set to **`true`** and the computer policy `Error1` might have the `Enabled` property set to **`false`**. For more information about policy merge issues, see the Merge user and computer policies section.

After creating the policies, why do policies get deleted when viewing policies in the Citrix Studio?

Empty policies are not allowed in the database to ensure efficiency when policies are evaluated by VDAs. Empty policies are policies without settings. The only exception is the `Unfiltered` policy, which is a built-in policy. It is impossible to create an empty policy in the Citrix Studio. But it is possible to create an empty policy using the provider. Some people might have scripts to create some empty policies temporarily. But if the policies are viewed using Citrix Studio, those policies are deleted and will not be visible in the Citrix Studio. To avoid having your temporary empty policies deleted, do not use the Citrix Studio to view policies while changes are being made.

How to retrieve the policy updates that is done in the Citrix Studio?

You or another user might be modifying policies at the same time. For example, you might have a PowerShell session and also a Citrix Studio session running at the same time. Changes made using the provider are saved to the database immediately (unless `AutoWriteBack` is set to **`false`**). Your changes made through the provider are available after you refresh the Citrix Studio view.

If you update a policy in the Citrix Studio and you want to pick it up in your current PowerShell session, you can use the `Refresh()` command.

```
1 PS GP:\User> (Get-PSDrive GP).Refresh()
```

Getting started with the SDK

March 11, 2024

To create a script, perform the following steps:

1. Use Citrix Studio to perform the operation that you want to script; for example, to create a catalog for a set of Machine Creation Services Machines.
2. Collect the log of SDK operations that Studio made to perform the task.
3. Review the script to understand what each part is doing. This will help you with the customization of your own script. For more information, see the example use case which explains in detail what the script is doing.
4. Convert and adapt the Studio script fragment to turn it into a script that is more consumable. To do this:

- Use variables. Some cmdlets take parameters, such as TaskId. However, it may not be clear where the value used in these parameters comes from because Studio uses values from the result objects from earlier cmdlets.
- Remove any commands that are not required.
- Add some steps into a loop so that these can be easily controlled. For example, add machine creation into a loop so that the number of machines being created can be controlled.

Examples

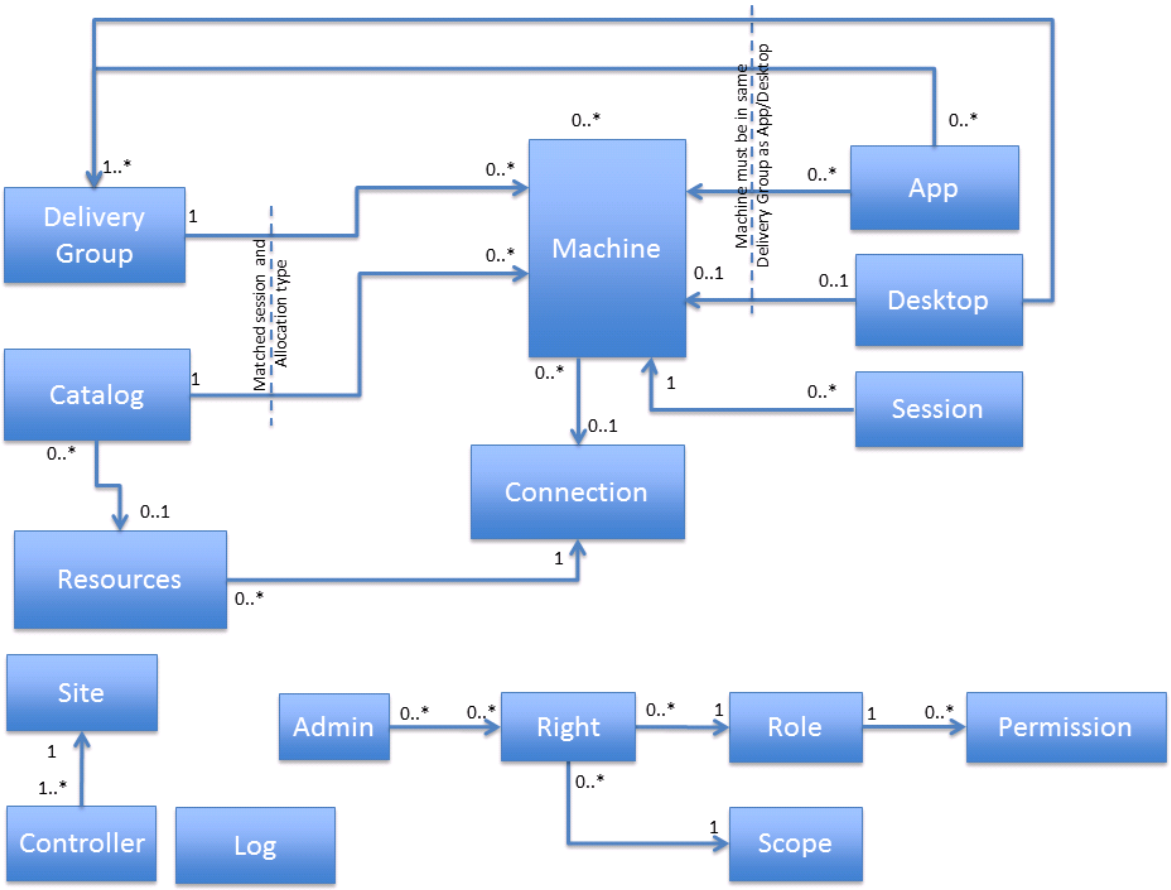
> **Note:**
>
> When creating a script, to ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described above rather than copying and pasting the example scripts.

Examples	Description
Example: Create catalog	Script: create a catalog for a set of Machine Creation Services (MCS) machines
Example: Create and configure a host	Script: create and configure a hosting connection
Example: Create a PvD Desktop	Script: create a Delivery Group containing static desktops
Example: Get load balancing information	Display load index values for Server OS Machines

Understanding the Citrix Virtual Apps and Desktops Administration Model

March 11, 2024

Citrix Virtual Apps and Desktops has an administration model that is defined by a set of objects and their interconnections, as shown in the following illustration.



The following sections describe each object, its responsibilities, characteristics and basic properties.

Connection

A Connection entity provides the details required to establish a connection to the administration point for a virtualization platform. These details are used to provide any power management and provisioning functions that are required. The Connection defines:

- The identity of the connection, which includes a name and an internal ID value (a GUID).
- Type —XenServer, SCVMM or vCenter.
- Username —the account name to use for connections to the hypervisor.
- Password —the password to use for connections to the hypervisor.
- The state of the connection and whether the hypervisor can be communicated with or not.
- Any current alerts raised by the hypervisor.

The password cannot be read back once set and is stored in the database in an encrypted form.

There is one of these entities in the model for each platform definition; however, there will be many connections established to the virtualization platform at run time to service the requirements of the Citrix Virtual Apps and Desktops deployment. These connections can be shared by many Resources entities (see below for further details).

> **Note:**
>
> When setting up a Connection with Citrix Hypervisor, you specify a host or Citrix Hypervisor pool; when setting up a Connection with Microsoft Hyper-V or VMware, you provide details to the management servers (SCVMM or vCenter). Also, when configuring a Connection, you must specify the user name in the correct format, depending on the virtualization solution you're using. If you're setting up a Connection with XenServer, specify a plain user name; for example "username"; for a Connection with VMware or Hyper-V, specify a domain user name; for example "domain\username".

Resources

> **Note:**
>
> This applies only to the creation of new machines when using Machine Creation Services (MCS) or Provisioning Services (PVS).

This setting defines a set of resources within the virtualization platform that are available for the product to consume. This allows the administrator to constrain deployment to a subset of the resources that the platform provides, and enables differing sets of resources to be defined for different purposes. This can control the storage and networking usage of machines provisioned within the site. The Resources entity defines:

- Networks—the networks which are able to be used when provisioning new machines into the virtualization plaform (for PVS, this is the streaming network; only one can be chosen or the first one is used).
- Storage—the storage to use when provisioning new machines into the virtualization platform.

At least one Resources setting must be defined for each Connection object if a provisioning mechanism is to be used. Many Resources settings can be linked to a single Connection.

The connection details are provided by the Connection entity.

> **Note:**
>
> When specifying Resources on a Connection to either SCVMM or vCenter, you select the host or cluster as well as the storage and networking.

Administrator

This defines the person who can administer the product. There can be various administrators of the product, each with a different set of capabilities within the product. The Administrator defines:

- The AD account the administrator is defined by, defined as their name and SID value. This can be an individual user or a security group.
- The rights the administrator has (i.e. what roles are available over which scopes).

Machine

A machine is a representation of a virtual or physical computer that can be used within a site to provide sessions to users. The machine defines:

- Identity of the machine, such as SAM name, DNS name, host (hypervisor) name, IP address, SID, License ID etc.
- Status of the machine, including power state, registration state, Personal vDisk state, load index etc. A summary state value aggregates many of these into a single state value.
- Information about the environment and configuration of the machine, such as version numbers of installed operating systems and Citrix components, including the 'functional level' of the machine.
- Data relating to the most recent activity of the machines, such as the last reason for de-registering, the last power action performed on the machine, the last connection failure etc.
- Maintenance mode and 'WindowsConnectionSetting' states for controlling the enable/disable/drain behavior of the machine.
- Visible user-resource settings for the machine for the assigned desktop case, such as icon, published name etc.
- For multi-session machines (Server OS Machines), aggregate information about the sessions running, such as the number of sessions active, pending etc.
- 'Tag' values associated with an assigned machine.
- Provisioning image information, such as the path to the master image, the provisioning scheme used to create the machine, and whether or not the machine has an updated image pending on its next reboot.

Most of the machine values are exposed by the Broker Service SDK, but items are also exposed by other services, such as the Machine Creation Service (MCS). Other values shown on a per-machine basis are inherited from the Catalog or Delivery Group (if any) that the machine is associated with, or

from the provisioning scheme used to create it. For example, the machine type
which is defined at the Catalog level by a combination of factors such Physical or Virtual, static or random, how user changes are persisted and others. For single-session machines (Desktop OS Machines), information about the session (if any) that is running on the machine is also associated with the machine. This includes connected user identity, session state, protocol in-use and so on. Machines can exist only in the model if they are defined as part of a Catalog; they cannot exist
outside of this concept.

Catalog

A Catalog defines a set of machines that are usually, but not always, expected to be equivalent. Multiple Catalogs can exist within a single deployment, enabling different sets of machines to be built and stored for different purposes. The Catalog defines:

- Catalog type, defined by the values of the following properties:
    1. The provisioning method for the machines (MCS, PVS or manual)
    2. How machines in the Catalog are allocated to users: statically with permanent assignment to users or randomly each time a user requests a resource
    3. Whether the machines are single-session (Desktop OS Machines) or multi-session (Server OS Machines)
    4. Whether the machines are physical or virtual
    5. How user changes to machines are handled, whether they are discarded after the user logs off or preserved locally on the machine or using Personal vDisk
- If the Catalog is to be used for Remote PC users and, if so, which Active Directory (AD) OUs are to be associated with the Catalog, and which Delivery Groups are associated with the Catalog.
- MCS provisioning-associated details (if MCS provisioning is to be used):
    1. Master image for the machines
    2. Memory size and number of CPUs
    3. Personal vDisk disk size, drive letter, and allocation percentages
    4. AD account naming scheme and OU where the machines are created, and a list of already created accounts
- Details of any PVS server associated with the Catalog.
- The functional level expected for machines; machines of lower functional level are not allowed to register with the site.

Also exposed at the Catalog level are some usage values for machines consumed or available in the Catalog.

Delivery Group

The Delivery Group provides details about a collection of machines used to provide desktops and/or applications to an end-user. Many Delivery Groups can be linked to the same Catalog, enabling machines in a Catalog to be distributed in various ways depending on the requirements of different user sets. The Delivery Group defines:

- The allocation type of the machines in the group, indicating whether machines are shared between users (random) or assigned persistently to one or more users (static), and whether the machines are single-session (Desktop OS) or multi-session (Server OS) machines.
- The delivery type of the group, indicating whether the group serves applications only, desktops only, or a mixture of applications and desktops to users.
- Settings controlling the power management of machines in the group, including:
    1. Which hours of the day, on different days of the week, are considered 'peak' time
    2. How many machines to keep running at different hours of the day on different days of the week for random/unassigned machines, including buffer sizes
    3. The timezone to use for evaluating the hours of the day for the above settings
    4. Whether and how assigned machines are power managed
    5. Whether and how to shut down or suspend machines after trigger events, such as user disconnect or logoff
    6. Whether or not the machines are considered corrupted by any sessions run on them and forced to restart to return them back to a clean, known state after each use
- How desktop resources from the group appear to an end-user, including icon used, color depth and name. Also defined is the security level required on the ICA connections for machines in the group. For desktop resources, the number of desktops each user is allowed simultaneously from the group.
- Rules used to establish the availability of these machines to the end users. Rules can factor in, not just the user's identity, but also where the user is connecting from and how, what the state of the client device is, and which remoting protocols are supported.
- Whether the group is enabled or disabled, including setting of maintenance mode.
- Whether the end-user is allowed to reset the machine themselves (for example, using StoreFront).

- 'Tag'values associated with a random/shared group.
- The functional level expected for machines in the group; machines of lower functional level are not allowed to register with the site.
- Whether the group is to be used for Remote PC users, and which Remote PC catalogs are associated with the group.
- A schedule for regularly rebooting multi-session (Server OS) machines at a particular time and day, and settings to control how that reboot is done.

Also associated with Delivery Groups are settings for features such as Profile management and Storefront URL settings, which are separately defined as 'Machine Configuration'objects and associated with one or more Delivery Groups.

> **Note:** The SDK often uses the term 'Desktop Group'to refer to a 'Delivery Group'

Application

This provides details of a seamless (i.e. floating window, separate from a desktop) application that is to be made available to end-users.Typically, each application is associated with a single delivery group, but application definitions can be shared across multiple groups, if required.

Applications can be run either on a remote machine and displayed on the local client desktop, or installed and run on the local client machine with windows overlaid onto a remote desktop. The application defines:

- The type of the application, whether 'HostedOnDesktop'or 'InstalledOnClient'.
- For HostedOnDesktop applications:
    1. The path to the application initial executable to be run on the VDA machine and the command line parameters, if any, to be supplied when the application is started
    2. Optionally, a specific set of users who have access to the application as a subset of users who have access to the Delivery Group(s)
    3. Any application-specific settings to be applied to the application process, including a CPU priority level, whether the application should wait for proxy printers to be created or not, etc.
- For InstalledOnClient applications:
    1. A flag to indicate that the icon for the application is to be fetched from the client device
    2. A flag to specify that extra security measures are to be taken with the arguments supplied to the application

- How the application resource appears to an end-user, including icon used, folder location on the client device, name and whether the shortcut appears in the start menu, desktop or both.
- Any file-type associations for the application, associating file extensions with the application.
- Which Delivery Group(s) it is associated with, along with an optional priority value for choosing between multiple groups.
- Whether the application is enabled and, separately, whether it is visible to end users or not.

Desktop

In Citrix Virtual Apps and Desktops, the Desktop object (which describes both the machine and the session on the machine) is replaced by the Session object and the Machine object, both of which have been expanded to do the work of the Desktop object.

Session

This provides details of a Windows session running on a machine controlled by the site. The session may be one initiated by Citrix Virtual Apps and Desktops, or one that was created by other means, such as a user logging directly onto the machine through the console or over RDP.

The session defines:

- The identity of the machine where the session is running, including machine DNS name, IP address, NetBios name, SID etc.
- The identity of the user who is running the session, including SAM name, UPN, SID, etc.
- The identity of the user who brokered the connection to the session, including SAM name, SID, etc.
- The identity of the endpoint client machine being used to connect to the session, including the device name, IP address, ID.
- The identity of the machine used to request the launch, i.e. the web server from which the launch was made. This includes name and IP address.
- The identity of the machine used to act as a gateway for the session connection, including machine DNS name, IP address.
- Details of significant events in the session, including start time, the time when the session was most recently connected to, brokered to etc.
- The durations of aspects of the most recent session creation or connection, including the time taken to broker the session, time taken to create the session etc.

- The current status of the session, including an overall session state, whether the ICA connection is secured, what protocol is being used.
- The current state of the machine running the session, incuding an overall summary state, power state, etc.
- Details about how the session connection was made, such as whether the session was brokered or connected to autonomously, the session context 'Smart Access'tags in force.
- Whether the session is 'hidden'or not. Sessions can become hidden if certain types of problems are encountered when a user launches an application or desktop.
- The list of brokered applications executing in the session.

The session SDK object also provides information from the related machine, Delivery Group and Catalog SDK objects. This information includes identity information, basic configuration and status information.

Controller

This provides details of the individual Delivery Controller machines in the site. Most of the data is dynamic state data from the running site, rather than configuration settings. The controller shows:

- Which 'site services'are active on each controller.
- The version of the controller components.
- The identity of the controller machine, as fully qualified DNS name, SAM name, SID etc.
- The type and version of the OS of the controller machine.
- Current controller and service status, and most recent activity times.
- Counts of machines registered with the controller.
- Which hypervisor connections are associated with the controller for site service location purposes.

Most of the controller values are exposed from the Broker Service SDK, but other items are exposed from other services, such as the Machine Creation Service.

Site

The Site is a top-level, logical representation of the Citix Virtual Apps and Desktops site, from the perspective of the configuration services running within it. The site contains licensing information, site metadata and the site name, among others.

A Citix Virtual Apps and Desktops installation has only a single configuration site instance. The Site object has the following properties:

- Name
- Controllers —list of controllers in the site
- Databases —list of databases used by the site
- DefaultIconUid
- LicenseInformation
- Metadata

Log

This provides details of the collected configuration logs, which describe the administrator activity on the site since logging was enabled. Administrator read actions are not logged, but any administrator action that changes the configuration or state of the site is included in the log. You can view the log at one of three levels: high-level logs, low-level logs and operation details. Each low-level log is a part of a larger high-level operation which is logged, while operation details describe elements within a single low-level operation. Log items show:

- The identity of the administrator who performed the operation, including the IP address of the machine from which it was performed.
- When the operation took place, both start and end times.
- Whether the operation succeeded or failed for any reason.
- A description of the operation, as a text string and also characterized by operation type, source type, target type etc.
- Details of parameters supplied to the operation.
- Any parent/child relationship in the log hierarchy.
- For changes in configuration, a before and after value for the items changed.

For more information about what is logged, see the Configuration Logging documentation.

Right

This defines a combination of role and the scope over which role permissions are allowed. Permissions defined in the role can be executed by the specified administrator, but only on objects that are directly or indirectly associated with the specified scope. The Right defines:

- The role whose permissions are to be allowed.
- The scope over which the role permissions are to be allowed.

Rights are not SDK objects that can be manipulated separately; they are always associated with a particular Administrator object. A single administrator can have many rights, with the total capability of the administrator being the sum of all their individual rights.

Scope

This defines a named grouping for objects, where objects in the grouping have administrator rights over objects controlled on a role-by-role basis. SDK objects of various types can have scopes directly associated with them, such as Catalogs, Delivery Groups, Connections, Resources etc. These SDK objects have properties that list the scopes that the object has been associated with. Other SDK objects have administrator rights granted by a secondary association with other objects which are directly scoped. For example, Machine objects inherit their scope associations from the Catalogs and Delivery Groups they are members of; Session objects inherit their scope associations from the machines on which the session is running. Some scopes are pre-defined (in practice only the 'All' scope is currently built-in) but you can create other scopes to specify suitable grouping definitions for your particular deployment.

The Scope defines:

- The identity of the scope, which includes a name and an internal ID value (a GUID).
- Whether the scope is built-in or not.

Role

Defines a set of permissions that an administrator can perform. Roles are always granted to administrators with an associated scope; they do not provide rights on their own, although some roles may have general permissions that apply to objects which do not have any scopes associated with them ('unscoped objects'). Some roles are pre-defined, but you can create other custom roles to specify suitable sets of permissions for your
particular deployment. A Role defines:

- The identity of the role, which includes a name and an internal ID value (a GUID)
- Whether the role is built-in or not
- The set of permissions that make up the role

The built-in roles are:

- Full Administrator—can perform all tasks and operations.
- Read Only Administrator—can see all objects in specified scopes, as well as global information, but cannot change anything.

- Machine Catalog Administrator—can create and manage Machine Catalogs and provision machines.
- Delivery Group Administrator—can deliver applications, desktops, and machines; can also manage the associated sessions. Allows creation and managing of Delivery Groups and applications.
- Help Desk Administrator—can view Delivery Groups, and manage the sessions and machines associated with those groups. Allows viewing of end-user resources and limited state change actions for troubleshooting end-user problems, but does not allow most configuration changes.
- Host Administrator—can manage host connections and their associated resource settings.

Permission

Defines a single console-level task or operation that is allowed when the permission is included in a role. Each permission can allow several low-level SDK operations (cmdlets), and a particular low-level SDK operation can be granted by any number of related permissions. A Permission defines:

- The identity of the permision, which includes a name and an internal ID value (a GUID).
- The permission group membersip of the permission. Permission groups collect together permissions relating to a particular functional area.
- The set of low-level SDK operations covered by the permission.

Creating a Catalog

March 11, 2024

The following example shows how to create a catalog for a set of Machine Creation Services (MCS) machines.

Before you begin, make sure you follow the steps detailed in Get started with the SDK.

This document tells you how to use Studio to perform the operation you want to script (in this case, to create a catalog for a set of Machine Creation Services machines) and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating catalog creation.

> **Note:**
>
> To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this document, rather than copying and pasting the example script. Line

> breaks have been added to the script for readability.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers have been added for readability.

```
1 Start-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.
     com:80' -Source 'Studio' -StartTime 29/05/2023 14:43:08 `
2     -Text 'Create Machine Catalog `'ExampleMachines`''
```

Starts a logged operation and returns a log ID which is supplied to subsequent operations to associate them with the larger task.

```
1 New-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
     AllocationType 'Permanent' -Description 'Example Machines' `
2     -IsRemotePC $False -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
        MinimumFunctionalLevel 'L7' -Name 'ExampleMachines' `
3     -PersistUserChanges 'OnPvd' -ProvisioningType 'MCS' -Scope @() -
        SessionSupport 'SingleSession'
```

Creates a Broker catalog. This catalog is populated with machines which are about to be created.

```
1 New-AcctIdentityPool -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
     -AllowUnicode -Domain 'dumdev.internal.citrix.com' `
2     -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97
        c7-397cc1238e46 -NamingScheme 'Example-####' `
3     -NamingSchemeType 'Numeric' -OU 'OU=DUMVMs,DC=dumdev,DC=internal,DC=
        citrix,DC=com' -Scope @()
```

Creates an Identity Pool. This defines the mechanism for creating AD computer accounts. This becomes a container for AD accounts created for the machines that are to be created.

```
1 Set-BrokerCatalogMetadata -AdminAddress 'ddc.dumdev.internal.citrix.com
     :80' -CatalogId 1 `
2     -LoggingId 39a2792-064a-43eb-97c7-397cc1238e46 -Name '
        Citrix_DesktopStudio_IdentityPoolUid' `
3     -Value 'b99aee6d-8772-4dbc-978b-8eb9a26e2407'
```

Sets metadata on the Broker catalog with details of the Identity Pool. This is not essential.

```
1 Test-ProvSchemeNameAvailable -AdminAddress 'ddc.dumdev.internal.citrix.
     com:80' -ProvisioningSchemeName @('ExampleMachines')
```

Checks that the requested name is available. This is not essential.

```
1 New-ProvScheme -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
     CleanOnBoot -HostingUnitName 'SharedNFS' `
```



```
    -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97
       c7-397cc1238e46 `
    -MasterImageVM 'XDHyp:\hostingunits\SharedNFS\BaseVM.vm\Base OS,
       domain joined and activated.snapshot \Pre-reqs installed.snapshot
       \\Updates Applied.snapshot\VDA75-no agent.snapshot\Updated Agent.
       snapshot' `
    -NetworkMapping @{
 0='xdhyp:\hostingunits\SharedNFS\Network0.network' }
  -ProvisioningSchemeName 'ExampleMachines' `
   -RunAsynchronously -Scope @() -UsePersonalVDiskStorage -VMCpuCount 1
       -VMMemoryMB 1024
```

Creates a provisioning scheme object. This is a template for the machines that are to be created. It specifies the hypervisor, network, storage, memory, number of CPUs to be used etc. It takes parameters from the system already set up, such as the HostingUnit name and the path to the VM snapshot to be used for the machines to be created. This command makes a 'consolidated'copy of the VM snapshot being used and, as a result, the process can take time to complete.

In this example, the Studio script specified the -RunAsyncronous flag on this command. This means the command will return control to the administrator before it has completed, so you must wait for it to finish before performing any operations that require it to be complete. If this flag is not specified, the command runs synchronously in-line and control is not returned until the command completes (successfully or otherwise). You can check the status of an asynchronous task using the Get-ProvTask cmdlet. Supply the task ID returned from the operation that started the task; in this case, the New-ProvScheme cmdlet.

```
Set-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
   LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 `
    -Name 'ExampleMachines' -ProvisioningSchemeId 76125e3a-9001-4993-86
       b6-eefc85c87880
```

Updates the BrokerCatalog with the unique Id of the provisioning scheme created above.

```
Add-ProvSchemeControllerAddress -AdminAddress 'ddc.dumdev.internal.
   citrix.com:80' `
    -ControllerAddress @('DDC.dumdev.internal.citrix.com') `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
       ProvisioningSchemeName 'ExampleMachines'
```

Adds a set of controller addresses to the provisioning scheme object. This is a list of addresses that the machines created can use to register with a Controller (broker) when deployed. The machines'registration addresses can be supplied in many ways; however, this information is required if the administrator wants to use the 'Allow Machine Creation Service to supply this'in the VDA installer. Changes to this list affect only machines created after the change, not existing machines.

```
Get-AcctADAccount -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
   IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 `
    -Lock $False -MaxRecordCount 2147483647 -State 'Available'
```


Studio gets a list of available Machine Identities from the Identity Pool so that, if existing accounts have been created in the past but are unused, these can be consumed instead of creating new accounts. Note that this is not required in a script because new accounts can be created instead, provided the script is running in a context that has permissions to do this. However, if the script does not have permissions to create accounts, change the script to consume available accounts (a separate process will be required to provide a pool of accounts into the Identity Pool, before running the script).

```
New-AcctADAccount -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    Count 2 `
    -IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 -LoggingId
       f39a2792-064a-43eb-97c7-397cc1238e46
```

Creates the required AD computer accounts in Active Directory. The script creates one account but, if required, it can create more using the 'Count'parameter of the command. The accounts are created into the OU defined in the provisioning scheme created above.

```
New-ProvVM -ADAccountName @('DUMDEV\Example-0001\$','DUMDEV\Example
    -0002\$') -AdminAddress 'ddc.dumdev.internal.citrix.com:80' `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
       ProvisioningSchemeName 'ExampleMachines' -RunAsynchronously
```

Creates virtual machines, based on the template definition in the provisioning scheme created above. This process may take time to complete.

```
Lock-ProvVM -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 `
     -ProvisioningSchemeName 'ExampleMachines' -Tag 'Brokered' -VMID @('
        0710bb77-d01f-d006-4d67-5472e5cd349f')
```

Locks the provisioned virtual machines and prevents accidental modification of the virtual machine. Consumers of the SDK can use this to indicate that the virtual machine is in use and why it is locked. The script locks the VM with a tag of 'Brokered'to indicate the virtual machine is created and added to a Broker catalog and must not be deleted without first being removed from the catalog. You can set the Tag name to whatever is required.

```
New-BrokerMachine -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    CatalogUid 1 `
    -HostedMachineId '0710bb77-d01f-d006-4d67-5472e5cd349f' `
    -HypervisorConnectionUid 1 -LoggingId f39a2792-064a-43eb-97c7-397
       cc1238e46 `
    -MachineName 'S-1-5-21-3918710733-2340574387-1999698698-109114'
```

Creates a Broker Machine object. These are objects stored in the catalog which join the provisioned machine with the catalog.

```
Start-BrokerMachinePvdImagePrepare -AdminAddress 'ddc.dumdev.internal.
    citrix.com:80' -InputObject @(2) `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46
```


Requests the Broker Service to initiate a preparation operation for Personal vDisk. This is required to allow the machine to initialize the storage for Personal vDisk.

```
Stop-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.com:80' `
   -HighLevelOperationId f39a2792-064a-43eb-97c7-397cc1238e46 -IsSuccessful $true
```

Stops the logged operation begun in the first step and indicates it was successful.

Customize the script

The following section shows how to convert and adapt the Studio output into a script that is more consumable. In addition to using variables and removing commands that are not required, it shows how to add machine creation into a loop so that you can control the number of machines created.

```
[CmdletBinding()]
param
(
    [Parameter(Mandatory=$true)] [string] $hostingUnitPath,
    [Parameter(Mandatory=$true)] [string] $catalogName,
    [string] $catalogDescription,
    [Parameter(Mandatory=$true)] [int] $numVmsToCreate,
    [string] $adminAddress,
    [Parameter(Mandatory=$true)] [string] $namingScheme,
    [string] $OU,
    [Parameter(Mandatory=$true)] [string] $domain,
    [Parameter(Mandatory=$true)] [string] $masterImagePath
)

Set-HypAdminConnection -AdminAddress $adminAddress

$hostingUnit = get-item $hostingUnitPath
$hostConnection = $hostingUnit.hypervisorConnection

$brokerHypConnection = Get-BrokerHypervisorConnection -HypHypervisorConnectionUid $hostConnection.HypervisorConnectionUid

# Start logged operation
$loggingOp = Start-LogHighLevelOperation -AdminAddress $adminAddress -Source 'Scripted' -Text "Create Machine Catalog `'$catalogName`'"
$loggingId = $loggingOp.Id

# Create the broker catalog and the AD Identity account pool
$catalog = New-BrokerCatalog -AllocationType 'Permanent' -Description $catalogDescription -IsRemotePC $False `
   -MinimumFunctionalLevel 'L7' -Name $catalogName -PersistUserChanges 'OnPvd' -ProvisioningType 'MCS' `
   -Scope @() -SessionSupport 'SingleSession' -LoggingId $loggingId -AdminAddress $adminAddress

```



```
$adPool = New-AcctIdentityPool -IdentityPoolName $catalogName -NamingScheme $namingScheme `
   -NamingSchemeType 'Numeric' -OU $OU -Domain $domain -AllowUnicode `
   -LoggingId $loggingId -AdminAddress $adminAddress

Set-BrokerCatalogMetadata -CatalogId $catalog.Uid -Name 'Citrix_DesktopStudio_IdentityPoolUid' `
   -Value $adPool.IdentityPoolUid -LoggingId $loggingId -AdminAddress $adminAddress

###########################################################################
# Create the ProvisioningScheme and wait for it to complete (reporting progress)

$provSchemeTaskID = New-ProvScheme -ProvisioningSchemeName $catalogName -HostingUnitUID $hostingUnit.HostingUnitUID `
   -IdentityPoolUID $adpool.IdentityPoolUid -CleanOnBoot -MasterImageVM $masterImagePath -UsePersonalVDiskStorage `
   -RunAsynchronously -LoggingId $loggingId -AdminAddress $adminAddress

$ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress $adminAddress
$taskProgress = 0

write-host "Creating New ProvScheme"
while ($provTask.Active -eq $true)
{

   # catch an uninitialized task progress, this occurs until the product initialized the value
   try {
 $totalPercent = if ($provTask.TaskProgress){
 $provTask.TaskProgress }
  else {
 0 }
  }
  catch {
  }

   Write-Progress -activity "Creating Provisioning Scheme:" -status "$totalPercent% Complete:" -percentcomplete $totalPercent
   sleep 30
   $ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress $adminAddress
 }

write-host "New ProvScheme Creation Finished"

$provScheme = get-provScheme -ProvisioningSchemeUID $provTask.ProvisioningSchemeUid
$controllers = Get-BrokerController | select DNSName
Add-ProvSchemeControllerAddress -ProvisioningSchemeUID $provScheme.ProvisioningSchemeUID -ControllerAddress $controllers `
```



```
72     -LoggingId $loggingId -AdminAddress $adminAddress
73
74 ###################################################################
75 # Set the provisioning scheme id for the broker catalog
76
77 Set-BrokerCatalog -InputObject $catalog -ProvisioningSchemeId $provTask
       .ProvisioningSchemeUid -LoggingId $loggingId -AdminAddress
       $adminAddress
78
79 ###################################################################
80 # create the AD accounts required and then create the Virtual machines
       (reporting progress)
81 $accts = New-AcctADAccount -IdentityPoolUid $adPool.IdentityPoolUid -
       Count $numVMsToCreate -LoggingId $loggingId `
82    -AdminAddress $adminAddress
83
84 $provVMTaskID = New-ProvVM -ProvisioningSchemeUID $provScheme.
       ProvisioningSchemeUID `
85    -ADAccountName $accts.SuccessfulAccounts -RunAsynchronously -
          LoggingId $loggingId -AdminAddress $adminAddress
86
87 # wait for the VMS to finish Provisioning
88 $ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress
       $adminAddress
89 while ($provTask.Active -eq $true)
90 {
91
92    # catch an uninitialized task progress, this occurs until the
          product initialized the value
93    try {
94  $totalPercent = if ($provTask.TaskProgress){
95  $provTask.TaskProgress }
96   else {
97  0 }
98   }
99   catch {
100  }
101
102   Write-Progress -activity "Creating Machines:" -status "$totalPercent
          % Complete:" -percentcomplete $totalPercent
103   sleep 5
104   $ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress
          $adminAddress
105 }
106
107
108 write-host "VM Creation Finished"
109
110 # Lock the VMs and add them to the broker Catalog
111 $provisionedVMs = get-ProvVM -ProvisioningSchemeUID
112 $provScheme.ProvisioningSchemeUID -AdminAddress $adminAddress
113
114 $provisionedVMs | Lock-ProvVM -ProvisioningSchemeUID $provScheme.
```



```
        ProvisioningSchemeUID `
115     -Tag 'Brokered' -LoggingId $loggingId -AdminAddress $adminAddress
116  $provisionedVMs | ForEach-Object {
117   New-BrokerMachine -CatalogUid $catalog.UID -HostedMachineId $_.VMId `
118     -HypervisorConnectionUid $brokerHypConnection.UID -MachineName $_.
            ADAccountSid -LoggingId $loggingId `
119     -AdminAddress $adminAddress }
120
121
122  Stop-LogHighLevelOperation -IsSuccessful $true -HighLevelOperationId
        $loggingId -AdminAddress $adminAddress
```

Configure support for non-domain joined catalogs

Using the Citrix Virtual Apps and Desktops service, you can create catalogs based on workgroup, or, non-domain joined machines. Creating non-domain joined machines depends on how the account identity pool is created. The account identity pool is the mechanism used by MCS to create and track machine names during catalog provisioning.

For example, in past releases all Active Directory fields were supplied in a single instance:

```
1  New-AcctIdentityPool AllowUnicode -Domain "awsdevexample.local" -
       IdentityPoolName "DedicatedHostCatalog" -NamingScheme "MH-DHost##" `
2      -NamingSchemeType "Numeric" *-OU "CN=Computers,DC= awsdevexample,DC=
           local"* -Scope @() `
3      -ZoneUid "81291221-d2f2-49d2-ab12-bae5bbd0df05"
```

MCS uses a new PoSH parameter, `WorkgroupMachine`, to create a workgroup catalog. Using the same example, noted above, this parameter removes the requirement to specify all the AD-specific parameters, including domain administrator credentials:

```
1  New-AcctIdentityPool AllowUnicode -WorkgroupMachine -IdentityPoolName "
       DedicatedHostCatalog" -NamingScheme "MH-DHOst##" `
2      -NamingSchemeType "Numeric" -Scope @() -ZoneUid "81291221-d2f2-49d2-
           ab12-bae5bbd0df05"
```

> **Note:**
>
> When using the WorkgroupMachine parameter, consider that non-domain joined machine catalogs are only supported through Powershell for all catalog lifecycle events including provisioning, adding/removing machines from the catalog, updating, and power management.

Creating a Catalog

March 11, 2024

The following example shows how to create a catalog for a set of Machine Creation Services (MCS) machines.

Before you begin, make sure you follow the steps detailed in Get started with the SDK.

This document tells you how to use Studio to perform the operation you want to script (in this case, to create a catalog for a set of Machine Creation Services machines) and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating catalog creation.

> **Note:**
>
> To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this document, rather than copying and pasting the example script. Line breaks have been added to the script for readability.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers have been added for readability.

```
1 Start-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.
     com:80' -Source 'Studio' -StartTime 29/05/2023 14:43:08 `
2     -Text 'Create Machine Catalog `'ExampleMachines`''
```

Starts a logged operation and returns a log ID which is supplied to subsequent operations to associate them with the larger task.

```
1 New-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
     AllocationType 'Permanent' -Description 'Example Machines' `
2     -IsRemotePC $False -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
        MinimumFunctionalLevel 'L7' -Name 'ExampleMachines' `
3     -PersistUserChanges 'OnPvd' -ProvisioningType 'MCS' -Scope @() -
        SessionSupport 'SingleSession'
```

Creates a Broker catalog. This catalog is populated with machines which are about to be created.

```
1 New-AcctIdentityPool -AdminAddress 'ddc.dumdev.internal.citrix.com:80'
     -AllowUnicode -Domain 'dumdev.internal.citrix.com' `
2     -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97
        c7-397cc1238e46 -NamingScheme 'Example-####' `
3     -NamingSchemeType 'Numeric' -OU 'OU=DUMVMs,DC=dumdev,DC=internal,DC=
        citrix,DC=com' -Scope @()
```

Creates an Identity Pool. This defines the mechanism for creating AD computer accounts. This becomes a container for AD accounts created for the machines that are to be created.


```
Set-BrokerCatalogMetadata -AdminAddress 'ddc.dumdev.internal.citrix.com
    :80' -CatalogId 1 `
    -LoggingId 39a2792-064a-43eb-97c7-397cc1238e46 -Name '
       Citrix_DesktopStudio_IdentityPoolUid' `
    -Value 'b99aee6d-8772-4dbc-978b-8eb9a26e2407'
```

Sets metadata on the Broker catalog with details of the Identity Pool. This is not essential.

```
Test-ProvSchemeNameAvailable -AdminAddress 'ddc.dumdev.internal.citrix.
    com:80' -ProvisioningSchemeName @('ExampleMachines')
```

Checks that the requested name is available. This is not essential.

```
New-ProvScheme -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    CleanOnBoot -HostingUnitName 'SharedNFS' `
    -IdentityPoolName 'ExampleMachines' -LoggingId f39a2792-064a-43eb-97
       c7-397cc1238e46 `
    -MasterImageVM 'XDHyp:\hostingunits\SharedNFS\BaseVM.vm\Base OS,
       domain joined and activated.snapshot \Pre-reqs installed.snapshot
       \\Updates Applied.snapshot\VDA75-no agent.snapshot\Updated Agent.
       snapshot' `
    -NetworkMapping @{
 0='xdhyp:\hostingunits\SharedNFS\Network0.network' }
  -ProvisioningSchemeName 'ExampleMachines' `
   -RunAsynchronously -Scope @() -UsePersonalVDiskStorage -VMCpuCount 1
       -VMMemoryMB 1024
```

Creates a provisioning scheme object. This is a template for the machines that are to be created. It specifies the hypervisor, network, storage, memory, number of CPUs to be used etc. It takes parameters from the system already set up, such as the HostingUnit name and the path to the VM snapshot to be used for the machines to be created. This command makes a 'consolidated' copy of the VM snapshot being used and, as a result, the process can take time to complete.

In this example, the Studio script specified the -RunAsyncronous flag on this command. This means the command will return control to the administrator before it has completed, so you must wait for it to finish before performing any operations that require it to be complete. If this flag is not specified, the command runs synchronously in-line and control is not returned until the command completes (successfully or otherwise). You can check the status of an asynchronous task using the Get-ProvTask cmdlet. Supply the task ID returned from the operation that started the task; in this case, the New-ProvScheme cmdlet.

```
Set-BrokerCatalog -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 `
    -Name 'ExampleMachines' -ProvisioningSchemeId 76125e3a-9001-4993-86
       b6-eefc85c87880
```

Updates the BrokerCatalog with the unique Id of the provisioning scheme created above.


```
Add-ProvSchemeControllerAddress -AdminAddress 'ddc.dumdev.internal.
    citrix.com:80' `
    -ControllerAddress @('DDC.dumdev.internal.citrix.com') `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
       ProvisioningSchemeName 'ExampleMachines'
```

Adds a set of controller addresses to the provisioning scheme object. This is a list of addresses that the machines created can use to register with a Controller (broker) when deployed. The machines' registration addresses can be supplied in many ways; however, this information is required if the administrator wants to use the 'Allow Machine Creation Service to supply this' in the VDA installer. Changes to this list affect only machines created after the change, not existing machines.

```
Get-AcctADAccount -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 `
    -Lock $False -MaxRecordCount 2147483647 -State 'Available'
```

Studio gets a list of available Machine Identities from the Identity Pool so that, if existing accounts have been created in the past but are unused, these can be consumed instead of creating new accounts. Note that this is not required in a script because new accounts can be created instead, provided the script is running in a context that has permissions to do this. However, if the script does not have permissions to create accounts, change the script to consume available accounts (a separate process will be required to provide a pool of accounts into the Identity Pool, before running the script).

```
New-AcctADAccount -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    Count 2 `
    -IdentityPoolUid b99aee6d-8772-4dbc-978b-8eb9a26e2407 -LoggingId
       f39a2792-064a-43eb-97c7-397cc1238e46
```

Creates the required AD computer accounts in Active Directory. The script creates one account but, if required, it can create more using the 'Count' parameter of the command. The accounts are created into the OU defined in the provisioning scheme created above.

```
New-ProvVM -ADAccountName @('DUMDEV\Example-0001\$','DUMDEV\Example
    -0002\$') -AdminAddress 'ddc.dumdev.internal.citrix.com:80' `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 -
       ProvisioningSchemeName 'ExampleMachines' -RunAsynchronously
```

Creates virtual machines, based on the template definition in the provisioning scheme created above. This process may take time to complete.

```
Lock-ProvVM -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    LoggingId f39a2792-064a-43eb-97c7-397cc1238e46 `
     -ProvisioningSchemeName 'ExampleMachines' -Tag 'Brokered' -VMID @('
        0710bb77-d01f-d006-4d67-5472e5cd349f')
```

Locks the provisioned virtual machines and prevents accidental modification of the virtual machine. Consumers of the SDK can use this to indicate that the virtual machine is in use and why it is locked. The script locks the VM with a tag of 'Brokered' to indicate the virtual machine is created and added to

a Broker catalog and must not be deleted without first being removed from the catalog. You can set the Tag name to whatever is required.

```
New-BrokerMachine -AdminAddress 'ddc.dumdev.internal.citrix.com:80' -
    CatalogUid 1 `
    -HostedMachineId '0710bb77-d01f-d006-4d67-5472e5cd349f' `
    -HypervisorConnectionUid 1 -LoggingId f39a2792-064a-43eb-97c7-397
       cc1238e46 `
    -MachineName 'S-1-5-21-3918710733-2340574387-1999698698-109114'
```

Creates a Broker Machine object. These are objects stored in the catalog which join the provisioned machine with the catalog.

```
Start-BrokerMachinePvdImagePrepare -AdminAddress 'ddc.dumdev.internal.
    citrix.com:80' -InputObject @(2) `
    -LoggingId f39a2792-064a-43eb-97c7-397cc1238e46
```

Requests the Broker Service to initiate a preparation operation for Personal vDisk. This is required to allow the machine to initialize the storage for Personal vDisk.

```
Stop-LogHighLevelOperation -AdminAddress 'ddc.dumdev.internal.citrix.
    com:80' `
    -HighLevelOperationId f39a2792-064a-43eb-97c7-397cc1238e46 -
       IsSuccessful $true
```

Stops the logged operation begun in the first step and indicates it was successful.

Customize the script

The following section shows how to convert and adapt the Studio output into a script that is more consumable. In addition to using variables and removing commands that are not required, it shows how to add machine creation into a loop so that you can control the number of machines created.

```
[CmdletBinding()]
param
(
    [Parameter(Mandatory=$true)] [string] $hostingUnitPath,
    [Parameter(Mandatory=$true)] [string] $catalogName,
    [string] $catalogDescription,
    [Parameter(Mandatory=$true)] [int] $numVmsToCreate,
    [string] $adminAddress,
    [Parameter(Mandatory=$true)] [string] $namingScheme,
    [string] $OU,
    [Parameter(Mandatory=$true)] [string] $domain,
    [Parameter(Mandatory=$true)] [string] $masterImagePath
)

Set-HypAdminConnection -AdminAddress $adminAddress

$hostingUnit = get-item $hostingUnitPath
```



```
$hostConnection = $hostingUnit.hypervisorConnection

$brokerHypConnection = Get-BrokerHypervisorConnection -
    HypHypervisorConnectionUid $hostConnection.HypervisorConnectionUid

# Start logged operation
$loggingOp = Start-LogHighLevelOperation -AdminAddress $adminAddress -
    Source 'Scripted' -Text "Create Machine Catalog `'$catalogName`'"
$loggingId = $loggingOp.Id

# Create the broker catalog and the AD Identity account pool
$catalog = New-BrokerCatalog -AllocationType 'Permanent' -Description
    $catalogDescription -IsRemotePC $False `
   -MinimumFunctionalLevel 'L7' -Name $catalogName -PersistUserChanges
       'OnPvd' -ProvisioningType 'MCS' `
   -Scope @() -SessionSupport 'SingleSession' -LoggingId $loggingId -
       AdminAddress $adminAddress

$adPool = New-AcctIdentityPool -IdentityPoolName $catalogName -
    NamingScheme $namingScheme `
   -NamingSchemeType 'Numeric' -OU $OU -Domain $domain -AllowUnicode `
   -LoggingId $loggingId -AdminAddress $adminAddress

Set-BrokerCatalogMetadata -CatalogId $catalog.Uid -Name '
    Citrix_DesktopStudio_IdentityPoolUid' `
   -Value $adPool.IdentityPoolUid -LoggingId $loggingId -AdminAddress
       $adminAddress

###########################################################################
# Create the ProvisioningScheme and wait for it to complete (reporting
    progress)

$provSchemeTaskID = New-ProvScheme -ProvisioningSchemeName $catalogName
     -HostingUnitUID $hostingUnit.HostingUnitUID `
   -IdentityPoolUID $adpool.IdentityPoolUid -CleanOnBoot -MasterImageVM
       $masterImagePath -UsePersonalVDiskStorage `
   -RunAsynchronously -LoggingId $loggingId -AdminAddress $adminAddress

$ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress
    $adminAddress
$taskProgress = 0

write-host "Creating New ProvScheme"
while ($provTask.Active -eq $true)
{

   # catch an uninitialized task progress, this occurs until the
       product initialized the value
   try {
 $totalPercent = if ($provTask.TaskProgress){
 $provTask.TaskProgress }
  else {
 0 }
```



```
  }
  catch {
  }

   Write-Progress -activity "Creating Provisioning Scheme:" -status "
      $totalPercent% Complete:" -percentcomplete $totalPercent
   sleep 30
   $ProvTask = get-provTask -TaskID $provSchemeTaskID -AdminAddress
      $adminAddress
 }

write-host "New ProvScheme Creation Finished"

$provScheme = get-provScheme -ProvisioningSchemeUID $provTask.
   ProvisioningSchemeUid
$controllers = Get-BrokerController | select DNSName
Add-ProvSchemeControllerAddress -ProvisioningSchemeUID $provScheme.
   ProvisioningSchemeUID -ControllerAddress $controllers `
   -LoggingId $loggingId -AdminAddress $adminAddress

#####################################################################
# Set the provisioning scheme id for the broker catalog

Set-BrokerCatalog -InputObject $catalog -ProvisioningSchemeId $provTask
   .ProvisioningSchemeUid -LoggingId $loggingId -AdminAddress
   $adminAddress

#####################################################################
# create the AD accounts required and then create the Virtual machines
   (reporting progress)
$accts = New-AcctADAccount -IdentityPoolUid $adPool.IdentityPoolUid -
   Count $numVMsToCreate -LoggingId $loggingId `
   -AdminAddress $adminAddress

$provVMTaskID = New-ProvVM -ProvisioningSchemeUID $provScheme.
   ProvisioningSchemeUID `
   -ADAccountName $accts.SuccessfulAccounts -RunAsynchronously -
      LoggingId $loggingId -AdminAddress $adminAddress

# wait for the VMS to finish Provisioning
$ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress
   $adminAddress
while ($provTask.Active -eq $true)
{

   # catch an uninitialized task progress, this occurs until the
      product initialized the value
   try {
 $totalPercent = if ($provTask.TaskProgress){
 $provTask.TaskProgress }
  else {
 0 }
  }
```



```
  catch {
  }

   Write-Progress -activity "Creating Machines:" -status "$totalPercent
      % Complete:" -percentcomplete $totalPercent
   sleep 5
   $ProvTask = get-provTask -TaskID $provVMTaskID -AdminAddress
      $adminAddress
 }


write-host "VM Creation Finished"

# Lock the VMs and add them to the broker Catalog
$provisionedVMs = get-ProvVM -ProvisioningSchemeUID
$provScheme.ProvisioningSchemeUID -AdminAddress $adminAddress

$provisionedVMs | Lock-ProvVM -ProvisioningSchemeUID $provScheme.
   ProvisioningSchemeUID `
   -Tag 'Brokered' -LoggingId $loggingId -AdminAddress $adminAddress
$provisionedVMs | ForEach-Object {
 New-BrokerMachine -CatalogUid $catalog.UID -HostedMachineId $_.VMId `
   -HypervisorConnectionUid $brokerHypConnection.UID -MachineName $_.
      ADAccountSid -LoggingId $loggingId `
   -AdminAddress $adminAddress }


Stop-LogHighLevelOperation -IsSuccessful $true -HighLevelOperationId
   $loggingId -AdminAddress $adminAddress
```

Configure support for non-domain joined catalogs

Using the Citrix Virtual Apps and Desktops service, you can create catalogs based on workgroup, or, non-domain joined machines. Creating non-domain joined machines depends on how the account identity pool is created. The account identity pool is the mechanism used by MCS to create and track machine names during catalog provisioning.

For example, in past releases all Active Directory fields were supplied in a single instance:

```
New-AcctIdentityPool AllowUnicode -Domain "awsdevexample.local" -
   IdentityPoolName "DedicatedHostCatalog" -NamingScheme "MH-DHost##" `
   -NamingSchemeType "Numeric" *-OU "CN=Computers,DC= awsdevexample,DC=
      local"* -Scope @() `
   -ZoneUid "81291221-d2f2-49d2-ab12-bae5bbd0df05"
```

MCS uses a new PoSH parameter, `WorkgroupMachine`, to create a workgroup catalog. Using the same example, noted above, this parameter removes the requirement to specify all the AD-specific parameters, including domain administrator credentials:


```
New-AcctIdentityPool AllowUnicode -WorkgroupMachine -IdentityPoolName "
    DedicatedHostCatalog" -NamingScheme "MH-DHOst##" `
    -NamingSchemeType "Numeric" -Scope @() -ZoneUid "81291221-d2f2-49d2-
       ab12-bae5bbd0df05"
```

> **Note:**
>
> When using the WorkgroupMachine parameter, consider that non-domain joined machine catalogs are only supported through Powershell for all catalog lifecycle events including provisioning, adding/removing machines from the catalog, updating, and power management.

Creating a Delivery Group

March 11, 2024

This document provides an example of a script that creates a Delivery Group containing static desktops.

Before you begin, make sure you follow the steps detailed in the Getting started guide, which shows you how to use Studio to perform the operation you want to script and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating the task.

> **Note:**
>
> To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this document, rather than copying and pasting the example script.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line numbers and line breaks have been added to the script for readability.

```
Start-LogHighLevelOperation -AdminAddress 'test-ddc.mydomain.com:80' -
    Source 'Studio' `
     -StartTime 31/07/2023 10:08:58 -Text 'Create Delivery Group "
        Windows 11 Desktops"'
```

Starts a logged operation and returns a log ID which is supplied to subsequent operations to associate them with the wider task.

```
New-BrokerDesktopGroup -AdminAddress 'test-ddc.mydomain.com:80' -
    ColorDepth 'TwentyFourBit' `
```



```
    -DeliveryType 'DesktopsOnly' -DesktopKind 'Private' -
        InMaintenanceMode $False -IsRemotePC $False `
    -LoggingId 846f2d42-a994-4bce-ab58-be05c8d73b99 -
        MinimumFunctionalLevel 'L7' `
    -Name 'Windows 11 Desktops' -OffPeakBufferSizePercent 10 -
        PeakBufferSizePercent 10 `
    -PublishedName 'Windows 11 Desktops' -Scope @() -SecureIcaRequired
        $False `
    -SessionSupport 'SingleSession' -ShutdownDesktopsAfterUse $False -
        TimeZone 'GMT Standard Time'
```

Creates a new Delivery Group with options collected by the Studio wizard.

```
Add-BrokerMachinesToDesktopGroup -AdminAddress 'test-ddc.mydomain.com
    :80' `
    -Catalog 'Windows 11' -Count 2 -DesktopGroup 'Windows 11 Desktops'
        `
    -LoggingId 846f2d42-a994-4bce-ab58-be05c8d73b99
```

Adds the number of machines requested from the nominated catalog to the new Delivery Group.

```
Set-Variable -Name 'brokerUsers' -Value @('S
    -1-5-21-3291547628-200264090-930806513-1104','S
    -1-5-21-3291547628-200264090-930806513-1105')
Get-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Filter {
 (SID -in $brokerUsers) }
  -MaxRecordCount 2147483647
Remove-Variable -Name 'brokerUsers'
New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN
    \user1'
New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN
    \user2'
```

The above commands are not required, Studio is verifying users.

```
Test-BrokerAssignmentPolicyRuleNameAvailable -AdminAddress 'test-ddc.
    mydomain.com:80' -Name @('Windows 11 Desktops')
```

Studio checks that the policy assignment name is available to use.

```
New-BrokerAssignmentPolicyRule -AdminAddress 'test-ddc.mydomain.com:80'
     -DesktopGroupUid 41 -Enabled $True `
    -IncludedUserFilterEnabled $False -LoggingId 846f2d42-a994-4bce-
        ab58-be05c8d73b99 -MaxDesktops 1 `
    -Name 'Windows 11 Desktops'
```

Create the new policy assignment rule for the Delivery Group. No users are specified here so all control is through the access policy rule.

```
Set-Variable -Name 'brokerUsers' -Value @('S
    -1-5-21-3291547628-200264090-930806513-1104','S
    -1-5-21-3291547628-200264090-930806513-1105')
Get-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Filter {
```



```
 (SID -in $brokerUsers) }
  -MaxRecordCount 2147483647
Remove-Variable -Name 'brokerUsers'
New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN
   \user1'
New-BrokerUser -AdminAddress 'test-ddc.mydomain.com:80' -Name 'MYDOMAIN
   \user2'
```

The above commands are not required, Studio is performing further checks.

```
Test-BrokerAccessPolicyRuleNameAvailable -AdminAddress 'test-ddc.
   mydomain.com:80' -Name @('Windows 11 Desktops\_Direct')
```

Studio tests that the access policy rule name is available to use.

```
New-BrokerAccessPolicyRule -AdminAddress 'test-ddc.mydomain.com:80' -
   AllowedConnections 'NotViaAG' `
    -AllowedProtocols @('HDX','RDP') -AllowRestart $True -
       DesktopGroupUid 41 -Enabled $True `
    -IncludedSmartAccessFilterEnabled $True -IncludedUserFilterEnabled
       $True `
    -IncludedUsers @('MYDOMAIN\user1','MYDOMAIN\user2') `
    -LoggingId 846f2d42-a994-4bce-ab58-be05c8d73b99 -Name 'Windows 11
       Desktops_Direct'
```

Creates the access policy rule for the new desktop for non-NetScaler Gateway connections.

```
Test-BrokerAccessPolicyRuleNameAvailable -AdminAddress 'test-ddc.
   mydomain.com:80' -Name @('Windows 11 Desktops_AG') New-
   BrokerAccessPolicyRule -AdminAddress 'test-ddc.mydomain.com:80' -
   AllowedConnections 'ViaAG' `
    -AllowedProtocols @('HDX','RDP') -AllowRestart $True -
       DesktopGroupUid 41 -Enabled $True `
    -IncludedSmartAccessFilterEnabled $True -IncludedSmartAccessTags @
       () -IncludedUserFilterEnabled $True `
    -IncludedUsers @('MYDOMAIN\user1','MYDOMAIN\user2') -LoggingId 846
       f2d42-a994-4bce-ab58-be05c8d73b99 -Name 'Windows 11 Desktops_AG'
```

Studio repeats this process for NetScaler Gateway connections.

```
Test-BrokerPowerTimeSchemeNameAvailable -AdminAddress 'test-ddc.
   mydomain.com:80' -Name @('Windows 11 Desktops_Weekdays') New-
   BrokerPowerTimeScheme -AdminAddress 'test-ddc.mydomain.com:80' -
   DaysOfWeek 'Weekdays' -DesktopGroupUid 41 `
    -DisplayName 'Weekdays' -LoggingId 846f2d42-a994-4bce-ab58-
       be05c8d73b99 -Name 'Windows 11 Desktops_Weekdays' `
    -PeakHours @($False,$False,$False,$False,$False,$False,$False,$True
       ,$True,$True,$True,$True,$True,$True, $True,$True,$True,
       $True,$False,$False,$False,$False,$False) `
    -PoolSize @(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
Test-BrokerPowerTimeSchemeNameAvailable -AdminAddress 'test-ddc.
   mydomain.com:80' -Name @('Windows 11 Desktops_Weekend')
New-BrokerPowerTimeScheme -AdminAddress 'test-ddc.mydomain.com:80' -
```



```
    DaysOfWeek 'Weekend' -DesktopGroupUid 41 `
 7   -DisplayName 'Weekend' -LoggingId 846f2d42-a994-4bce-ab58-
        be05c8d73b99 -Name 'Windows 11 Desktops_Weekend' `
 8   -PeakHours @($False,$False,$False,$False,$False,$False,$False,$True
        ,$True,$True,$True,$True,$True,$True,$True,$True,$True,$True,
        $True,$False,$False,$False,$False,$False) `
 9   -PoolSize @(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
```

Studio checks that the names for the (optional) weekday and weekend power schemes are available, and adds these.

```
1 Stop-LogHighLevelOperation -AdminAddress 'test-ddc.mydomain.com:80' -
    EndTime 31/07/2023 10:09:05 `
2   -HighLevelOperationId 846f2d42-a994-4bce-ab58-be05c8d73b99 -
        IsSuccessful $True
```

Stops the logged operation, indicating it was successful.

Customize the script

This section shows how to convert and adapt the Studio output into a script that is more consumable.

The script creates a Delivery Group containing Windows 11 desktops. The catalog specified in the parameters must exist already and be populated appropriately (with an allocation type of static). The script is designed to be run from a Powershell command line logged on as a Citrix administrator. No checks are made for permissions; the script will fail if the user does not have the appropriate permissions.

```
 1 <#
 2 Sample usage:
 3 .\CreateDeliveryGroup.ps1 `
 4     -GroupName "Windows 11 Desktops" `
 5     -SrcCatalog "win11-catalog" `
 6     -NumDesktops 2 ``
 7     -Users @('mydomain\user1','mydomain\user2')
 8 #>
 9
10 Param(
11     [Parameter(Mandatory=$true)] [string] $GroupName,
12     [Parameter(Mandatory=$true)] [string] $SrcCatalog,
13     [Parameter(Mandatory=$true)] [int] $NumDesktops,
14     [Parameter(Mandatory=$true)] [array] $Users
15     [string] $AdminAddress
16 )
```

The table explains the parameters used in the script.

Parameter	Description
SrcCatalog	The name of the catalog to be used to create the desktop. Create the catalog by specifying an allocation type of static.
GroupName	The name of the catalog to be used to create the desktop. Create the catalog by specifying an allocation type of static.
NumDesktops	The number of machines to add to the desktop group. If insufficient machines are available, as many as possible are added.
Users	Which users can access the group. This is a list of users or groups; for example, @('mydomain\Domain Users') or @('mydomain\user1','mydomain\user2')

```
Set-HypAdminConnection -AdminAddress $adminAddress
```

Specify the hypervisor admin connection to use. Removes the need for the -AdminAddress for some of the commands.

```
$peakPoolSize = 2
$weekendPoolSizeByHour = new-object int[] 24
$weekdayPoolSizeByHour = new-object int[] 24
9..17 | %{
  $weekdayPoolSizeByHour[$_] = $peakPoolSize  }

$peakHours = (0..23 | %{
  $_ -ge 9 -and $_ -le 17  }
 )
```

This creates 24 element arrays with a 1 or a 0 in each entry. Use these to specify when peak hours are for the power schedules for the Delivery Groups. Elements 9 to 17 (hours starting 09:00 to 17:00) for weekdays are set to 1, others are left at 0. Two unassigned machines are powered up during peak times, if available.

```
$logId = Start-LogHighLevelOperation` -Text "Create desktop group" -
    Source "Create Desktop Group Script"
```

Start a new logged operation. This returns a log ID which is passed into subsequent operations to associate them with the create group task.

```
$grp = New-BrokerDesktopGroup `
    -DesktopKind 'Private' `
    -DeliveryType 'DesktopsOnly' `
```



```
    -LoggingId $logId.Id `
    -Name $GroupName `
    -PublishedName $GroupName `
    -SessionSupport 'SingleSession' `
    -ShutdownDesktopsAfterUse $False

$count = Add-BrokerMachinesToDesktopGroup `
    -Catalog $SrcCatalog `
    -Count $NumDesktops `
    -DesktopGroup \$GroupName `
    -LoggingId $logId.Id

"$count machines added to the desktop group"
```

Create the new Delivery Group, delivering private desktops. The catalog used must have been populated with suitable machines. PublishedName is the name seen by end users; the following uses the same name as the group name.

```
$null = New-BrokerAssignmentPolicyRule `
    -DesktopGroupUid $grp.Uid `
    -IncludedUserFilterEnabled $False `
    -LoggingId $logId.Id `
    -MaxDesktops 1 `
    -Name ($GroupName + '\_AssignRule')
```

Assigned desktops need an assignment policy. Disable user filter so that access is controlled entirely by access policy rules.

```
$null = New-BrokerAccessPolicyRule `
    -AllowedConnections 'NotViaAG' `
    -AllowedProtocols @('HDX','RDP') `
    -AllowRestart $True `
    -DesktopGroupUid $grp.Uid `
    -IncludedSmartAccessFilterEnabled $True `
    -IncludedUserFilterEnabled $True `
    -IncludedUsers $Users `
    -LoggingId $logId.Id `
    -Name ($GroupName + '_Direct')

$null = New-BrokerAccessPolicyRule `
    -AllowedConnections 'ViaAG' `
    -AllowedProtocols @('HDX','RDP') `
    -AllowRestart $True `
    -DesktopGroupUid $grp.Uid `
    -IncludedSmartAccessFilterEnabled $True `
    -IncludedSmartAccessTags @() `
    -IncludedUserFilterEnabled $True `
    -IncludedUsers $Users `
    -LoggingId $logId.Id `
    -Name ($GroupName + '_AG')
```

Specify any access restrictions: allow direct access using NetScaler Gateway, using HDX & RDP proto-

cols. The user can request the desktop be restarted, if necessary.

```
$null = New-BrokerPowerTimeScheme `
    -DaysOfWeek 'Weekdays' `
    -DesktopGroupUid $grp.Uid `
    -DisplayName 'Weekdays' `
    -LoggingId $logId.Id `
    -Name ($GroupName + '_Weekdays') `
    -PeakHours $peakHours `
    -PoolSize $weekdayPoolSizeByHour

$null = New-BrokerPowerTimeScheme `
    -DaysOfWeek 'Weekend' `
    -DesktopGroupUid $grp.Uid `
    -DisplayName 'Weekend' `
    -LoggingId $logId.Id `
    -Name ($GroupName + '_Weekend') `
    -PeakHours $peakHours `
    -PoolSize $weekendPoolSizeByHour
```

Optional: Specify power schedules.

```
Stop-LogHighLevelOperation -HighLevelOperationId $logId.Id -
   IsSuccessful $True
```

Stop configuration logging and indicate if successful or not.

Creating and configuring a hosting connection

March 11, 2024

The following example shows how to create and configure a hosting connection.

Before you begin, make sure you follow the steps detailed in the Getting started guide, which tells you how to use Studio to perform the operation you want to script (in this case, to create a host) and collect the log of SDK operations that Studio made to perform the task. This output can then be customized to produce a script for automating host creation.

> **Note:**
>
> To ensure you always get the latest enhancements and fixes, Citrix recommends you follow the procedure described in this document, rather than copying and pasting the example script. Line numbers and line breaks have been added to the script for readability.

Understand the script

The following section explains what each part of the script produced by Studio is doing. This will help you with the customization of your own script. Line breaks have been added for readability.

```
Get-LogSite -AdminAddress 'mycontroller.example.com:80'
```

Queries the configuration logging service to retrieve information about the site configuration.

```
Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
    `
    -Source 'Studio' -StartTime 14/08/2013 14:30:28 -Text 'Create
       Connection `'Example Citrix Hypervisor`''
```

Starts a high-level logging operation with the configuration logging operation within which the rest of the commands will exist. Returns a log ID which is supplied to subsequent operations.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the location of the Host Service that will be used by the configuration cmdlets. Because the Host Service exposes a PowerShell provider, not all of the cmdlets can take an address for the service so this cmdlet sets a default location.

```
New-Item -ConnectionType 'XenServer' -HypervisorAddress @('http://
   hypervisorhost1.example.com') `
    -LoggingId e355ce51-8cbb-400a-ae81-1fdc567239cb -Path @('XDHyp:\
       Connections\Example Citrix Hypervisor') `
    -Scope @() -Password ******** -UserName 'root'
```

Creates a connection to a Citrix Hypervisor host (hypervisorhost1.example.com). This is a non-persistent connection
and is available only to this PowerShell runspace.

```
Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
   -EndTime 14/08/2013 14:30:29 `
    -HighLevelOperationId 'e355ce51-8cbb-400a-ae81-1fdc567239cb' -
       IsSuccessful $True
```

Stops the logged operation begun previously and indicates it was successful.

```
Get-LogSite -AdminAddress 'mycontroller.example.com:80'
```

Queries the configuration logging service to retrieve information about the site configuration.

```
Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
    -Source 'Studio' -StartTime 14/08/2013 14:30:30 `
    -Text 'Update Connection `'Example Citrix Hypervisor`''
```

Starts a new high-level logging operation.


```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the Host Service address details again (note that this repetition is removed in the optimized script below).

```
Set-Item -HypervisorAddress @('http://hypervisorhost1.example.com','
    http://hypervisorhost2.example.com') `
 -LoggingId 44e15629-6906-4840-a36c-984aaf67be6d -PassThru -Path @('
    XDHyp:\\Connections\\Example Citrix Hypervisor') `
 -Password ******** -UserName 'root'
```

Updates the connection previously created. As there is more than one Citrix Hypervisor server in the pool, it supplies all the addresses to enable High Availability.

```
Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
    -EndTime 14/08/2013 14:30:31 `
 -HighLevelOperationId '44e15629-6906-4840-a36c-984aaf67be6d' -
    IsSuccessful \$True
```

Stops the logging operation.

```
Get-LogSite -AdminAddress 'mycontroller.example.com:80'
```

Queries the configuration logging service to retrieve information about the site configuration.

```
Start-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
     -Source 'Studio' `
 -StartTime 14/08/2013 14:31:03 `
 -Text 'Create Resources `'Example Resources`' and Persist
    Connection `'Example Citrix Hypervisor`''
```

Starts a new logging operation.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the Host Service address details again.

```
Get-ChildItem -Path @('XDHyp:\Connections')
```

Gets the contents of the host connection to populate the wizard dialogs.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the Host Service address details again.

```
Remove-Item -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -Path @('
    XDHyp:\Connections\Example Citrix Hypervisor')
```

Removes the temporary connection created in the wizard.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```


Sets the Host Service address details again.

```
New-Item -ConnectionType 'XenServer' -HypervisorAddress @('http://
    hypervisorhost1.example.com','http://hypervisorhost2.example.com') `
     -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 -Path @('XDHyp:\\
        Connections\\Example Citrix Hypervisor') -Persist `
     -Scope @() -Password ******** -UserName 'root'
```

Recreates the connection as a persistent connection which is written to the database and available to other PowerShell runspaces.

```
New-BrokerHypervisorConnection -AdminAddress 'mycontroller.example.com
    :80' `
     -HypHypervisorConnectionUid a14096ba-5074-44ff-b596-371e345c0449 `
     -LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314
```

Adds the host connection to the Broker Service.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the Host Service address details again.

```
New-Item -HypervisorConnectionName 'Example Citrix Hypervisor' -
    LoggingId 76caa3f4-df93-4cb2-b78d-6a8824766314 `
     -NetworkPath @('XDHyp:\Connections\Example Citrix Hypervisor\
        Network0.network') `
     -Path @('XDHyp:\HostingUnits\Example Resources') `
     -RootPath 'XDHyp:\Connections\Example Citrix Hypervisor' `
     -StoragePath @('XDHyp:\Connections\Example Citrix Hypervisor\
        Primary OS.storage')
```

Creates the HostingUnit (referred to as Resources in Studio) using the information gathered in step 14.

```
Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'
```

Sets the Host Service address details again.

```
Get-Item -Path @('XDHyp:\Connections\Example Citrix Hypervisor')
```

Retrieves the newly created object.

```
Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80'
    -EndTime 14/08/2013 14:31:07 `
     -HighLevelOperationId '76caa3f4-df93-4cb2-b78d-6a8824766314' -
        IsSuccessful $True
```

Stops the logged operation begun previously and indicates if it was successful.

Customize the script

The following section shows how to convert and adapt the Studio output into a script that is more consumable. The following script has been simplified so that, instead of creating a temporary host connection in the process of acquiring information in the wizards as in the Studio script above, a persistent connection is created. Information is then queried from within this to create the HostingUnit (Resources). Note that the LoggingId and HypHyperConnectionUid details are different.

Line breaks have been added for readability.

```
$loggingOp = Start-LogHighLevelOperation -AdminAddress $adminAddress -Source 'Scripted' -Text "Create Connection '$connectionName'"
$loggingId = $loggingOp.Id

Set-HypAdminConnection -AdminAddress 'mycontroller.example.com:80'

New-Item -ConnectionType 'XenServer' -HypervisorAddress @('http://hypervisorhost1.example.com','http://hypervisorhost2.example.com')`
    -LoggingId $loggingId -Path @('XDHyp:\\Connections\\Example Citrix Hypervisor') -Persist -Scope @() `
    -Password 'XenServerPassword' -UserName 'root'

Get-ChildItem -Path @('XDHyp:\Connections')

New-BrokerHypervisorConnection -AdminAddress 'mycontroller.example.com:80' `
    -HypHypervisorConnectionUid a14096ba-5074-44ff-b596-371e345c0449 `
    -LoggingId $loggingId

New-Item -HypervisorConnectionName 'Example Citrix Hypervisor' -LoggingId $loggingId `
    -NetworkPath @('XDHyp:\Connections\Example Citrix Hypervisor\Network0.network') `
    -Path @('XDHyp:\HostingUnits\Example Resources') `
    -RootPath 'XDHyp:\Connections\Example Citrix Hypervisor' `
    -StoragePath @('XDHyp:\Connections\Example Citrix Hypervisor\PrimaryOS.storage')

Stop-LogHighLevelOperation -AdminAddress 'mycontroller.example.com:80' -HighLevelOperationId $loggingId -IsSuccessful $True
```

Getting load balancing information

March 11, 2024

You can use Server OS Machines to deliver cost-effective applications and desktops hosted on server operating systems to multiple users.

To load balance Server OS Machines in a deployment, you use Citrix policies. There are several load balancing policy settings for enabling and configuring load management between servers delivering Windows Server OS machines. For more information, see the load management policy settings reference documentation. You work with policies through Studio or the Group Policy Management Console in Windows; see the Policies documentation for details.

To see the load, you can use either the Citrix Director or Studio consoles, or the PowerShell SDK. The following example shows how to use the PowerShell SDK to display the load.

> **Note:**
>
> If you've used previous versions of Citrix Virtual Apps and Desktops, you may be familiar with the **qfarm/load** command. This tool is no longer available, but you can use PowerShell to display similar output as shown in the example below.

Example: Get load index values

To display a list of machines with their calculated/measured load index values, together with counts of sessions running on them:

1. Start a shell in PowerShell. For more information, see Citrix Virtual Apps and Desktops SDK.
2. Type:

```
Get-BrokerMachine -SessionSupport MultiSession -Property 'DnsName','
    LoadIndex','SessionCount'
```

> **Note:**
>
> Load index values go up to 10000. They indicate VDA machine load calculated from the configured sources, such as number of sessions. A value of 10000 indicates a fully loaded VDA machine; the broker will not send another user session to that machine.

For more information and examples, see the cmdlet help for the Get-BrokerMachine cmdlet and About topics such as about_broker_filtering.

about_AcctADIdentitySnapIn

March 11, 2024

Topic

about_AcctADIdentitySnapin

Short Description

The Active Directory Identity Service PowerShell snap-in provides administrative functions for the Active Directory Identity Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Acct'.

Long Description

The Active Directory Identity Service PowerShell snap-in enables both local and remote administration of the Active Directory Identity Service. It provides facilities to store details about Active Directory computer accounts that the Machine Creation Service can use.

The snap-in provides two main entities:

Identity

A representation of an Active Directory computer account that reflects the state of the account within the context of the Machine Creation Service. When an account is created by or imported into the Active Directory Identity Service, the account password is stored. Once the account is consumed by the Machine Creation Service, the password is discarded. For accounts registered with the Active Directory Identity Service, identities hold the following additional state information.

Available

The Active Directory account is registered with the service, the password for the account is known, and the account is available to be consumed by another service. Accounts that are successfully created with the New-AcctADAccount command or imported using the Add-AcctADAccount command are initially assigned this state.

InUse

The Active Directory account is registered and has been consumed by another service. The password for the account is no longer known to the service.

Error

The Active Directory account is registered, but is missing, disabled, or locked within Active Directory. Accounts that are not successfully created with the New-AcctADAccount command or imported using the Add-AcctADAccount command appear in this state. Use the Update-AcctADAccount and Repair-AcctADAccount commands to resolve issues with accounts in this state.

Tainted

The Active Directory account is registered and has been released by all the consuming services, but cannot be made available for use as the password is no longer known. Use the Repair-AcctADAccount command to reset account passwords and restore the account state to 'Available'.

Identities can also be marked as 'Locked'by the Machine Creation Service to indicate that they are in use and must not be changed. These services are also responsible for unlocking the Active Directory accounts when they no longer require exclusive access. In some cases, an account may remain locked if a task is unexpectedly stopped. Use the Unlock-AcctADAccount command to manually unlock the account if necessary.

Identity Pool

Containers for identities that can be configured with all the information required for new Active Directory accounts to be created. Alternatively, identity pools can be populated by importing accounts that already exist in Active Directory. All accounts registered with the Active Directory Identity Service must be placed into one of these containers. An identity can belong to more than one identity pool, but the state of the identity cannot be different in each pool. For example, an identity that is in use will be marked 'InUse'in all the identity pools of which it is part.

To avoid conflicting changes, identity pools can also be marked as 'Locked' during operations that modify the content of a pool. These operations are also responsible for unlocking the identity pool. In some cases, a pool may remain locked if a task is unexpectedly stopped. Use the Unlock-AcctIdentityPool command to manually unlock the pool if necessary.

Active Directory Permissions

Certain commands provided by the Active Directory Identity Service PowerShell snap-in require permissions for Active Directory operations. By default, the user account executing the command will be used to check for permissions. Active Directory credentials for an account with permissions may also be specified using the -ADUsername and -ADPassword parameters if necessary.

Account Creation (using the New-AcctADAccount command)

To use PowerShell to create new Active Directory accounts, the command must be run using an account with sufficient permissions in the required Active Directory container (specified by the identity pool organizational unit parameter) for accounts to be created.

Import Accounts (using the Add-AcctADAccount command)

There are two modes for this operation: situations where the Active Directory account passwords are known and situations where the passwords are not known.

If the account passwords are known, the accounts can be imported without the need for administrative permissions in Active Directory. The accounts are imported and the password provided is used to change the existing password.

If the passwords are not known, the command must be run using an account that has permissions to reset the password for the accounts.

Account Removal (using the Remove-AcctADAccount command)

By default, the Remove-AcctADAccount command will only remove accounts from the AD Identity Service database without needing permissions to modify Active Directory accounts.

An optional parameter 'RemovalOption'can be set to modify the Active Directory accounts. If set to 'Disable'or 'Delete', the command must be run with sufficient permissions to disable or delete the account from Active Directory.

about_AcctADIdentitySnapIn

March 11, 2024

Topic

about_AcctADIdentitySnapin

Short Description

The Active Directory Identity Service PowerShell snap-in provides administrative functions for the Active Directory Identity Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Acct'.

Long Description

The Active Directory Identity Service PowerShell snap-in enables both local and remote administration of the Active Directory Identity Service. It provides facilities to store details about Active Directory computer accounts that the Machine Creation Service can use.

The snap-in provides two main entities:

Identity

A representation of an Active Directory computer account that reflects the state of the account within the context of the Machine Creation Service. When an account is created by or imported into the Active Directory Identity Service, the account password is stored. Once the account is consumed by the Machine Creation Service, the password is discarded. For accounts registered with the Active Directory Identity Service, identities hold the following additional state information.

Available

The Active Directory account is registered with the service, the password for the account is known, and the account is available to be consumed by another service. Accounts that are successfully created with the New-AcctADAccount command or imported using the Add-AcctADAccount command are initially assigned this state.

InUse

The Active Directory account is registered and has been consumed by another service. The password for the account is no longer known to the service.

Error

The Active Directory account is registered, but is missing, disabled, or locked within Active Directory. Accounts that are not successfully created with the New-AcctADAccount command or imported using the Add-AcctADAccount command appear in this state. Use the Update-AcctADAccount and Repair-AcctADAccount commands to resolve issues with accounts in this state.

Tainted

The Active Directory account is registered and has been released by all the consuming services, but cannot be made available for use as the password is no longer known. Use the Repair-AcctADAccount command to reset account passwords and restore the account state to 'Available'.

Identities can also be marked as 'Locked' by the Machine Creation Service to indicate that they are in use and must not be changed. These services are also responsible for unlocking the Active Directory accounts when they no longer require exclusive access. In some cases, an account may remain locked if a task is unexpectedly stopped. Use the Unlock-AcctADAccount command to manually unlock the account if necessary.

Identity Pool

Containers for identities that can be configured with all the information required for new Active Directory accounts to be created. Alternatively, identity pools can be populated by importing accounts that already exist in Active Directory. All accounts registered with the Active Directory Identity Service must be placed into one of these containers. An identity can belong to more than one identity pool, but the state of the identity cannot be different in each pool. For example, an identity that is in use will be marked 'InUse' in all the identity pools of which it is part.

To avoid conflicting changes, identity pools can also be marked as 'Locked' during operations that modify the content of a pool. These operations are also responsible for unlocking the identity pool. In some cases, a pool may remain locked if a task is unexpectedly stopped. Use the Unlock-AcctIdentityPool command to manually unlock the pool if necessary.

Active Directory Permissions

Certain commands provided by the Active Directory Identity Service PowerShell snap-in require permissions for Active Directory operations. By default, the user account executing the command will be used to check for permissions. Active Directory credentials for an account with permissions may also be specified using the -ADUsername and -ADPassword parameters if necessary.

Account Creation (using the New-AcctADAccount command)

To use PowerShell to create new Active Directory accounts, the command must be run using an account with sufficient permissions in the required Active Directory container (specified by the identity pool organizational unit parameter) for accounts to be created.

Import Accounts (using the Add-AcctADAccount command)

There are two modes for this operation: situations where the Active Directory account passwords are known and situations where the passwords are not known.

If the account passwords are known, the accounts can be imported without the need for administrative permissions in Active Directory. The accounts are imported and the password provided is used to change the existing password.

If the passwords are not known, the command must be run using an account that has permissions to reset the password for the accounts.

Account Removal (using the Remove-AcctADAccount command)

By default, the Remove-AcctADAccount command will only remove accounts from the AD Identity Service database without needing permissions to modify Active Directory accounts.

An optional parameter 'RemovalOption'can be set to modify the Active Directory accounts. If set to 'Disable'or 'Delete', the command must be run with sufficient permissions to disable or delete the account from Active Directory.

about_Acct_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying

the name with an ordered list of elements or their numeric values,
or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and
supports combination of conditions with -and and -or, and grouping
with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use
script blocks in most cases. This can be easier to read as it reduces
quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double
quotes, and omit the quotes altogether for many strings. The order of
parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell
handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape
the wildcards using the normal PowerShell escape mechanisms (see
about_Escape_Characters), or switch to a filter expression and the -eq
operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
```



```
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}

By their nature, floating point values, DateTime values, and TimeSpan

values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```


Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid


```
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-AcctADAccount

March 11, 2024

Import existing Active Directory (AD) computer accounts for use in the AD Identity Service.

Syntax

```
Add-AcctADAccount
   [-IdentityPoolName] <String>
   -ADAccountName <String[]>
   [-Password <String>]
   [-SecurePassword <SecureString>]
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AcctADAccount
   -IdentityPoolUid <Guid>
   -ADAccountName <String[]>
   [-Password <String>]
   [-SecurePassword <SecureString>]
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Provides the ability for Active Directory (AD) computer accounts that already exist in AD to be used by the Citrix AD Identity Service and other Citrix Machine Creation Services.

All aspects of this command that need to make modifications to the accounts in AD will do so using the account that the PowerShell runspace is using. This means that if the passwords need resetting for the accounts, the user performing the operation in PowerShell must have sufficient privileges in AD for this operation to complete successfully. If the current user does not have sufficient privileges, the ADUserName and ADPassword parameters may be used instead.

The following rules apply to the importing of AD accounts:

If the current account password is supplied, the cmdlet will attempt to change the password so that it is known only to the Citrix Identity Service. This uses password change operations and does not need AD account administration permissions.

If the current password is not supplied, the cmdlet will attempt to reset the password for the AD account so that it is known only to the Citrix Identity Service. This requires the user to have enough privileges in AD to reset the account passwords.

Imported accounts in a disabled or locked state in AD are imported with the account marked in an error state.

If the identity pool into which accounts are being imported does not have a domain set, the domain of the identity pool will be set to the domain of the first account imported.

Examples

EXAMPLE 1

Import the two accounts (account and account2) from AD into the identity Pool called “MyPool”

```
Add-AcctADAccount -IdentityPoolName MyPool -ADAccountName "Domain\
    account","Domain\account2" -OutVariable result

SuccessfulAccounts                SuccessfulAccountsCount
    FailedAccountsCount    FailedAccounts
------------------                -----------------------
    -------------------    --------------
{
 domain\account, domain\account2 }
  2                             0                      {
  }


$result.SuccessfulAccounts

```



```
ADAccountGuid           : 473f5534-48ac-4baf-831f-376c5b813b60
ADAccountName           : domain\account
ADAccountSid            : S-1-5-21-1315084875-1285793635-2418178940-2644
AccountDisabled         : False
AccountLocked           : False
Domain                  : Domain.com
DomainControllerHint    :
Lock                    : False
State                   : Available
TenantId                :
DeviceManagementType    : None
IdentityType            : ActiveDirectory
VdaHostId               :
WorkgroupMachine        : False
TrustServiceInstanceId  :

ADAccountGuid           : 7f23cdaa-072f-4eab-bca5-c2b18cd30c06
ADAccountName           : domain\account2
ADAccountSid            : S-1-5-21-1315084875-1285793635-2418178940-2645
AccountDisabled         : False
AccountLocked           : False
Domain                  : Domain.com
DomainControllerHint    :
Lock                    : False
State                   : Available
TenantId                :
DeviceManagementType    : None
IdentityType            : ActiveDirectory
VdaHostId               :
WorkgroupMachine        : False
TrustServiceInstanceId  :
```

Parameters

-IdentityPoolName

The name of the identity pool in which the accounts are to be imported.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ADAccountName

The name of the AD account to be imported. AD accounts are accepted in the following formats:

- Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com
- UPN format e.g `MyComputer@MyDomain.Com`
- Domain qualified e.g MyDomain\MyComputer

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool in which the accounts are to be imported.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Password

The current password for the computer account.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SecurePassword

The current password for the account (provided in a Secure String class).

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADUserName

The username for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADPassword

The matching password for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	SecureString

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationDetailedSummary

The Add-AcctADAccount returns an object that contains the following properties:

- SuccessfulAccountsCount <int>

  The number of accounts that were added successfully

- FailedAccountsCount <int>

  The number of accounts that were not added.

- FailedAccounts <Citrix.XDPowerShell.AccountError[]>

  The list of accounts that failed to be added. Each one has the following properties:

  - ADAccountName <string>
  - ADAccountSid <String>
  - ErrorReason <ADIdentityStatus>

    This can be one of the following:

    * UnableToConvertDomain
    * UnableToAccessAccountProperties
    * IdentityNotLocatedInDomain
    * UnableToAccessAccountProperties
    * IdentityDuplicateObjectExists
    * IdentityObjectLocked
    * IdentityObjectInUse
    * FailedToConnectToDomainController
    * FailedToExecuteSearchInAD
    * FailedToAccessComputerAccountInAD
    * FailedToSetPasswordInAD
    * FailedToChangePasswordInAD
    * ADServiceDatabaseError
    * ADServiceDatabaseNotConfigured

        * ADServiceStatusInvalidDb
    - DiagnosticInformtion <Exception>

      Any other error information

- SuccessfulAccounts <Citrix.ADIdentity.Sdk.Identity[]>

  The list of accounts that were successfully added. Each one has the following properties:

    - ADAccountGuid <Guid>

      The unique identifier for the account.
    - ADAccountName <string>

      The name of the account.
    - ADAccountSid <string>

      The SID for the account.
    - AccountDisabled <bool>

      Whether or not the account is disabled in AD.
    - AccountLocked <bool>

      Whether or not the account is locked in AD.
    - Domain <string>

      The domain for the account.
    - DomainControllerHint <string>
    - Lock <bool>

      Whether or not the account is locked (in the database, not AD).
    - State <Citrix.ADIdentity.Sdk.ADIdentityState>

      The state for the account. This can be:

        * Available

          The account is not used.
        * InUse

          The account is in use.
        * Error

          The account is in error (i.e. the account is locked or disabled in AD).

    * Tainted

      The account is no longer used, but the password is no longer known.

  - TenantId <Guid>

    The identity of the tenant associated with this account.

  - DeviceManagementType <string>

    The device management type.

  - IdentityType <string>

    The identity type.

  - VdaHostId <Guid>

    The ID of the VDA associated with this account.

  - WorkgroupMachine <bool>

    Whether or not the account is a workgroup account (not domain-joined).

  - TrustServiceInstanceId <string>

    The trust service ID of the machine.

Notes

To maintain maximum security when using the command programmatically, Citrix recommends you use the 'SecurePassword'property instead of the 'Password'property.

In the case of failure, the following errors can result:

- IdentityPoolAlreadyLocked

  The specified identity pool was locked by another operation.

- IdentityPoolNotFound

  The specified identity pool was not found.

- IdentityDuplicateObjectExists

  The specified AD account already exists for the identity pool.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- ServiceStatusInvalidDb

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- CommunicationError

  An error occurred while communicating with the service.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Remove-AcctADAccount
- Repair-AcctADAccount
- Get-AcctADAccount
- Update-AcctADAccount
- Unlock-AcctADAccount

Add-AcctAzureADSecurityGroupMember

March 11, 2024

Adds an Azure AD security group to an assigned security group.

Syntax

```
Add-AcctAzureADSecurityGroupMember
   [-AccessToken] <String>
   [-GroupId] <String>
   [-RefGroupId] <String>
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to add a Citrix-created device security group as a member to an assigned security group.

Examples

EXAMPLE 1

Adds the security group of a device as a direct member to an assigned security group.

```
Add-AcctAzureADSecurityGroupMember -AccessToken $accessToken -GroupId
   $groupId -RefGroupId $refGroupId"
True
```

Parameters

-AccessToken

Access token of Microsoft Graph API. Make sure grant consent to following permissions:

- Group.ReadWrite.All
- GroupMember.ReadWrite.All

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-GroupId

The assigned security group's ID.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RefGroupId

The device security group's ID.

Type:	String
Position:	4
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

- AzureADTenantIdMismatchAzureADAccessToken

  Mismatch between the given AzureADTenantID and TenantID in Azure AD AccessToken.

Related Links

- about_AcctADIdentitySnapin

Add-AcctIdentityPoolScope

March 11, 2024

Add the specified identity pool(s) to the given scope(s).

Syntax

```
Add-AcctIdentityPoolScope
   [-Scope] <String[]>
   -InputObject <IdentityPool[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AcctIdentityPoolScope
   [-Scope] <String[]>
   -IdentityPoolUid <Guid[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AcctIdentityPoolScope
   [-Scope] <String[]>
   -IdentityPoolName <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-AcctIdentityPoolScope command is used to associate one or more identity pool objects with given scope(s).

To add an identity pool to a scope you need permission to change the scopes of the identity pool and permission to add objects to all of the scopes you have specified.

When attempting to add an identity pool to a scope it is already associated with, no change will be made.

The identity pools may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Adds a single identity pool to the 'Finance'scope.

```
Add-AcctIdentityPoolScope Finance -IdentityPoolUid 6702C5D0-C073-4080-
   A0EE-EC74CB537C52
```


EXAMPLE 2

Adds a single identity pool to the multiple scopes.

```
Add-AcctIdentityPoolScope Finance,Marketing -IdentityPoolUid 6702C5D0-
    C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 3

Adds all visible identity pool objects to the 'Finance'scope.

```
Get-AcctIdentityPool | Add-AcctIdentityPoolScope Finance
```

EXAMPLE 4

Adds identity pool objects with a name starting with an 'A'to the 'Finance'scope.

```
Add-AcctIdentityPoolScope Finance -IdentityPoolName A*
```

Parameters

-Scope

Specifies the scopes to add the objects to.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

The identity pool objects to be added.

Type:	IdentityPool[]

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pools to be added.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pools to be added.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctIdentityPoolScope
- Get-AcctScopedObject

Add-AcctServiceAccountScope

March 11, 2024

Add the specified service account(s) to the given scope(s).

Syntax

```
Add-AcctServiceAccountScope
   [-Scope] <String[]>
   -InputObject <ServiceAccount[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AcctServiceAccountScope
   [-Scope] <String[]>
   -ServiceAccountUid <Guid[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-AcctServiceAccountScope command is used to associate one or more service account objects with given scope(s).

To add a service account to a scope you need permission to change the scopes of the service account and permission to add objects to all of the scopes you have specified.

When attempting to add a service account to a scope it is already associated with, no change will be made.

The service accounts may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Adds a single service account to the 'Finance'scope.

```
Add-AcctServiceAccountScope -Scope Finance -ServiceAccountUid 17631afc
   -2e4c-491e-b0aa-f979a80e32c1
```


EXAMPLE 2

Adds a single service account to the multiple scopes.

```
Add-AcctServiceAccountScope Finance,Marketing -ServiceAccountUid 17631
    afc-2e4c-491e-b0aa-f979a80e32c1
```

EXAMPLE 3

Adds all visible service account objects to the 'Finance'scope.

```
Get-AcctServiceAccount | Add-AcctServiceAccountScope Finance
```

Parameters

-Scope

Specifies the scopes to add the objects to.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

The service account objects to be added.

Type:	ServiceAccount[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ServiceAccountUid

The unique identifier for the service accounts to be added.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctServiceAccountScope
- Get-AcctScopedObject

Copy-AcctIdentityPool

March 11, 2024

Copies an identity pool and any contained accounts to a new identity pool

Syntax

```
Copy-AcctIdentityPool
    [-IdentityPoolName] <String>
    [-NewIdentityPoolName] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Copy-AcctIdentityPool
    -IdentityPoolUid <Guid>
    [-NewIdentityPoolName] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Provides the ability to copy an identity pool.

The new identity pool will contain all the accounts that were in the original pool and will have the same domain and OU set. The naming scheme will be unset and the StartCount will be set to 1.

Examples

EXAMPLE 1

Copies the identity pool named "MyPool"to a new pool named "NewPool". Note that the NamingScheme is empty and the StartCount is set to 1.

```
Copy-AcctIdentityPool -IdentityPoolName MyPool -NewIdentityPoolName
    NewPool

AvailableAccounts    : 0
DeviceManagementType : None
Domain               : MyDomain.com
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : NewPool
IdentityPoolUid      : 2c94daf3-bf67-4e67-a7a2-d402e8f04e73
IdentityType         : ActiveDirectory
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         :
NamingSchemeType     : None
OU                   :
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
Scopes               :
TenantId             :
```

Parameters

-IdentityPoolName

The name of the identity pool that is to be copied.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool that is to be copied.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewIdentityPoolName

The name of the new identity pool.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

- AvailableAccounts <int>

  The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse' , 'Tainted', or 'Error').

- DeviceManagementType <string>

  The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

- Domain <string>

  The Active Directory domain (in FQDN format) that accounts in the pool belong to.

- ErrorAccounts <int>

  The number of existing AD accounts in the 'Error'state.

- IdentityContent <string>

  JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

- IdentityPoolName <string>

  The name of the identity pool.

- IdentityPoolUid <GUID>

  The unique identifier for the identity pool.

- IdentityType <string>

  The identity type.

- InUseAccounts <int>

  The number of existing AD accounts in the 'InUse'state.

- Lock <bool>

  Indicates if the identity pool is locked.

- MetadataMap <IDictionary[string, string];>

  The metadata associated with this identity pool arranged in key value pairs.

- NamingScheme <string>

  The naming scheme for the identity pool.

- NamingSchemeType <string>

  The naming scheme type for the identity pool. This can be one of the following:

  - Numeric

    Naming scheme uses numeric indexes.

- Alphabetic

  Naming scheme uses alphabetic indexes.

- OU <string>

  The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

- ResourceLocationId <GUID>

  The UID that corresponds to the resource location (DaaS only).

- StartCount <int>

  The next index to be used when creating an account in the identity pool.

- TaintedAccounts <int>

  The number of existing AD accounts in the 'Tainted' state.

- WorkgroupMachine <bool>

  If this is true, the identity pool can have an IdentityType of 'AzureAD' or 'Workgroup'.
  If this is false, the identity pool can have an IdentityType of 'ActiveDirectory' or 'HybridAzureAD'.

- ZoneUid <GUID>

  The UID that corresponds to the Zone in which AD accounts are created.

- Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

  The administration scopes associated with this identity pool.

- TenantId <GUID>

  Identity of the Citrix tenant associated with this identity pool.
  Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure, the following errors can result:

- IdentityPoolDuplicateObjectExists

  An identity pool with the same name exists already.

- IdentityPoolObjectNotFound

  The identity pool to be modified could not be located.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- ServiceStatusInvalidDb

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- CommunicationError

  An error occurred while communicating with the service.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Get-AcctIdentityPool
- Remove-AcctIdentityPool

Export-AcctIdentityPool

March 11, 2024

Exports an identity pool.

Syntax


```
Export-AcctIdentityPool
      [-IdentityPoolName] <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Export-AcctIdentityPool
      -IdentityPoolUid <Guid>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to export an identity pool and contained AD accounts to a JSON encoded string. Any accounts that have not yet been used by MCS will not be exported.

Examples

EXAMPLE 1

Exports the identity pool named "MyPool". This pool contains accounts that have not been used yet, so they are not exported.

```
Export-AcctIdentityPool -IdentityPoolName MyPool

{
 "IdentityPoolName":"MyPool","NamingScheme":"Acc###","NamingSchemeType"
    :2,"StartCount":7,"OU":"","Domain":"MyDomain.com","Lock":false,"
    TenantId":null,"ResourceLocationId":null,"ZoneUid":null,"
    WorkgroupMachine":false,"IdentityType":"ActiveDirectory","
    IdentityContent":null,"DeviceManagementType":"None","Accounts":[] }
```

EXAMPLE 2

Exports the identity pool named "MyPool". This pool contains accounts that have been used, so they are exported.

```
Export-AcctIdentityPool -IdentityPoolName MyPool

{
 "IdentityPoolName":"MyPool","NamingScheme":"Acc###","NamingSchemeType"
    :2,"StartCount":3,"OU":"","Domain":"MyDomain.com","Lock":false,"
    TenantId":null,"ResourceLocationId":"59e9d28d-f46b-434a-97c2-995
    da9e120d7","ZoneUid":"c5282cd0-6526-4f83-b1bd-6ea2d46a7c59","
    WorkgroupMachine":false,"IdentityType":"ActiveDirectory","
    IdentityContent":null,"DeviceManagementType":"None","Accounts":[{
```



```
5   "AccountSid":"S-1-5-21-4215727520-1727542857-3748865654-1404","
       AccountGuid":"85a82962-cbb2-4c74-aa2c-9502890e554b","Domain":"
       MyDomain.com","State":2,"AccountName":"MyDomain\\Acc001$","
       DomainControllerName":"
       v2_bnMtZGMubmljaG9sYXNzdS5sb2NhbDo1OWU5ZDI4ZC1mNDZiLTQzNGEtOTdjMi05OTVkYTllMT
       =","TimeCreated":"2022-10-20T16:48:04.55" }
6   ,{
7   "AccountSid":"S-1-5-21-4215727520-1727542857-3748865654-1405","
       AccountGuid":"984a0fb8-ef0d-49d3-8720-3602878bc843","Domain":"
       MyDomain.com","State":2,"AccountName":"MyDomain\\ExportAcc002$","
       DomainControllerName":"
       v2_bnMtZGMubmljaG9sYXNzdS5sb2NhbDo1OWU5ZDI4ZC1mNDZiLTQzNGEtOTdjMi05OTVkYTllMT
       =","TimeCreated":"2022-10-20T16:48:05.58" }
8   ] }
```

Parameters

-IdentityPoolName

The name of the identity pool.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-IdentityPoolUid

The unique identifier for the identity pool.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

A JSON encoded version of the identity pool and the ADAccounts that are part of the identity pool.

Notes

In the case of failure, the following errors can result:

- PartialData

  Only a subset of the available data was returned.
- CouldNotQueryDatabase

  The query required to get the database was not defined.
- PermissionDenied

  The user does not have administrative rights to perform this operation.
- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error
- CommunicationError

  An error occurred while communicating with the service.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Import-AcctIdentityPool
- Export-ProvScheme
- Import-ProvScheme

Get-AcctADAccount

March 11, 2024

Gets the Active Directory (AD) accounts stored in the AD Identity Service.

Syntax

```
Get-AcctADAccount
   [-ADAccountSid <String>]
   [-Domain <String>]
   [-IdentityPoolName <String>]
   [-State <ADIdentityState>]
   [-Lock <Boolean>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Get-AcctADAccount
    [-ADAccountSid <String>]
    [-Domain <String>]
    [-IdentityPoolUid <Guid>]
    [-State <ADIdentityState>]
    [-Lock <Boolean>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to locate the Active Directory (AD) accounts stored within the AD Identity Service and view the state of the accounts.

Examples

EXAMPLE 1

Return all the AD accounts that are registered in the AD Identity Service.

```
Get-AcctADAccount

ADAccountGuid          : a33f54f8-4944-4537-93c9-a04f0b889378
ADAccountName          : MyDomain\ACC001
ADAccountSid           : S-1-5-21-1315084875-1285793635-2418178940-2684
AccountDisabled        : False
AccountLocked          : False
Domain                 : MyDomain.com
DomainControllerHint   :
    v2_ZGMubXlkb21haW4uY29tOjU5ZTlkMjhkLWY0NmItNDM0YS05N2MyLTk5NWRhOWUxMjBkNw
    ==
Lock                   : False
State                  : Available
TenantId               :
DeviceManagementType   : None
IdentityType           : ActiveDirectory
VdaHostId              : ee3ec984-3f1b-41ed-aee7-38754692e829
WorkgroupMachine       : False
TrustServiceInstanceId : ee3ec984-3f1b-41ed-aee7-38754692e829-S
    -1-5-21-1315084875-1285793635-2418178940-2684

IdentityPoolName       : MyWorkgroupPool
```



```
IdentityPoolUid        : f4aef7af-4298-44a3-a5fb-4a9201ca01d7
ADAccountGuid          : 00000000-0000-0000-0000-000000000000
ADAccountName          : WorkgrpAcc001
ADAccountSid           : S
    -1-254-31435167-1163162762-1265062292-170227718-1001
AccountDisabled        : False
AccountLocked          : False
Domain                 :
DomainControllerHint   :
Lock                   : False
State                  : Available
TenantId               :
DeviceManagementType   : None
IdentityType           : Workgroup
VdaHostId              : 01dfa99f-748a-4554-9451-674b0678250a
WorkgroupMachine       : True
TrustServiceInstanceId : 01dfa99f-748a-4554-9451-674b0678250a
```

EXAMPLE 2

Return all the AD accounts that are registered in the AD Identity Service in the identity pool named "MyPool"that are not locked.

```
Get-AcctADAccount -IdentityPoolName MyPool -Lock $false
```

EXAMPLE 3

Return all the AD accounts that are registered in the AD Identity Service in the identity pool named "MyPool"or an identity pool with a name starting with 'p'. For full details of the advanced filtering aspects of this command see about_Acct_Filtering.

```
Get-AcctADAccount -Filter {
 IdentityPoolName -Like "p*" -or IdentityPoolName -eq "MyPool" }
```

Parameters

-ADAccountSid

The AD Account SID of the account.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Domain

The domain of the account (this is in dns format).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-State

The current state of the identity stored in the AD Identity Service for the AD account.

Type:	ADIdentityState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Lock

Indicates if the account is locked in the AD Identity Service.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool to which the account is registered.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-IdentityPoolUid

The unique identifier for the identity pool that the account is registered to.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityInPool

The Get-AcctADAccount returns an object that contains the following parameters:

- IdentityPoolName <string>

  The name of the containing identity pool.

- IdentityPoolUid <GUID>

  The unique identifier for the containing identity pool.

- ADAccountGuid <GUID>

  The unique identifier for the account.

- ADAccountName <string>

  The name of the account.

- ADAccountSid <string>

  The SID for the account.

- AccountDisabled <bool>

  Whether or not the account is disabled in AD.

- AccountLocked <bool>

  Whether or not the account is locked in AD.

- Domain <string>

  The domain for the account.

- DomainControllerHint <string>

  The base 64 encoded hint for the domain controller location.

- Lock <bool>

  Whether or not the account is locked (in the database, not AD).

- State <string>

  The state for the account. This can be:

- TenantId <GUID>

  The identity of the tenant associated with this account.

- DeviceManagementType <string>

  The device management type.

- IdentityType <string>

  The identity type.

- VdaHostId <GUID>

  The ID of the VDA associated with this account.

- WorkgroupMachine <bool>

  Whether or not the account is a workgroup account (not domain-joined).

- TrustServiceInstanceId <string>

  The trust service ID of the machine.

Notes

In the case of failure the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- CouldNotQueryDatabase

  The query required to get the database was not defined.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Add-AcctADAccount
- Remove-AcctADAccount
- Unlock-AcctADAccount
- Update-AcctADAccount
- Repair-AcctADAccount

Get-AcctAzureADSecurityGroup

March 11, 2024

Retrieve the security groups in specific Azure tenant.

Syntax


```
Get-AcctAzureADSecurityGroup
    [-AccessToken] <String>
    [-GroupId <String>]
    [-Name <String>]
    [-SearchString <String>]
    [-Assigned <Boolean>]
    [-Dynamic <Boolean>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to retrieve the Azure AD security groups of specific Azure tenant.

Examples

EXAMPLE 1

Gets Azure AD security groups with the object id of “e17d1c86-efgh-efgh-efgh-97e22c7bd96c”.

```
Get-AcctAzureADSecurityGroup -AccessToken $accessToken -GroupId "
    e17d1c86-efgh-efgh-efgh-97e22c7bd96c"

MembershipRule : (device.displayName -match "AzureADMC[0-9]{
 3 }
 $")
Name : SecurityGroupOfAzureADMachineCatalog
ObjectId  : e17d1c86-efgh-efgh-efgh-97e22c7bd96c
Type : Dynamic
```

EXAMPLE 2

Gets AzureAD security group with the name of “ABCD”.

```
Get-AcctAzureADSecurityGroup -AccessToken $accessToken -Name "
    SecurityGroupOfAzureADMachineCatalog"

MembershipRule : (device.displayName -match "AzureADMC[0-9]{
 3 }
 $")
```



```
Name : SecurityGroupOfAzureADMachineCatalog
ObjectId  : e17d1c86-efgh-efgh-efgh-97e22c7bd96c
Type : Dynamic
```

EXAMPLE 3

Gets AzureAD security group with name contains of “ABCD”.

```
Get-AcctAzureADSecurityGroup -AccessToken $accessToken -SearchString "
    SecurityGroup"

MembershipRule : (device.displayName -match "AzureADMC[0-9]{
 3 }
 $")
Name : SecurityGroupOfAzureADMachineCatalog
ObjectId  : e17d1c86-efgh-efgh-efgh-97e22c7bd96c
Type : Dynamic
```

EXAMPLE 4

Gets all AzureAD assigned security groups.

```
Get-AcctAzureADSecurityGroup -AccessToken $accessToken -AssignedOnly

MembershipRule : (device.displayName -match "AzureADMC[0-9]{
 3 }
 $")
Name : SecurityGroupOfAzureADMachineCatalog
ObjectId  : e17d1c86-efgh-efgh-efgh-97e22c7bd96c
Type : Assigned
```

Parameters

-AccessToken

Access token of Microsoft Graph API. Make sure grant consent to following permissions:

- Group.Read.All

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	True

-GroupId

The ObjectId of an AzureAD security group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Name

The Name of the AzureAD security group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SearchString

The search expression of the AzureAD security group display name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Assigned

Specify only assigned AzureAD security group will retrieve.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Dynamic

Specify only dynamic AzureAD security group will retrieve.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AzureADSecurityGroup

This object provides details of an object of the Azure AD security group and contains the following information:

- MembershipRule <string>

  The membershipRule of AzureAD security group.

- Name <string>

  The name of AzureAD security group.

- ObjectId <Guid>

  The unique Object Id of the AzureAD security group.

- Type <string>

  The type of AzureAD security group.

Notes

The parameter GroupId will take priority over other parameters.

When all parameters are empty, it will return all assigned security groups under current tenant id that encoded in assess token.

In the case of failure, the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs

Related Links

- about_AcctADIdentitySnapin

Get-AcctAzureADSecurityGroupMember

March 11, 2024

Retrieves all the security group members of a specific security group.

Syntax

```
Get-AcctAzureADSecurityGroupMember
    [-AccessToken] <String>
    [-GroupId] <String>
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to retrieve all the security group members from a specific Azure AD group.

Examples

EXAMPLE 1

Retrieves all the security group members of a specific Azure AD security group.

```
Get-AcctAzureADSecurityGroupMember -AccessToken $accessToken -
    AzureADTenantId $azureADTenantId -GroupId $gid -Top 10"

MembershipRule : (device.displayName -match "AzureADMC[0-9]{
 3 }
 $")
Name : SecurityGroupOfAzureADMachineCatalog
ObjectId  : e17d1c86-efgh-efgh-efgh-97e22c7bd96c
Type : Dynamic
```

Parameters

-AccessToken

Access token of Microsoft Graph API. Make sure grant consent to following permissions:

- Group.Read.All
- GroupMember.Read.All

Type: String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-GroupId

Specifies the ID of a group in Azure AD.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AzureADSecurityGroup

This object provides details of an object of the Azure AD security group and contains the following information:

- MembershipRule <string>

  The membershipRule of AzureAD security group.

- Name <string>

  The name of AzureAD security group.

- ObjectId <Guid>

  The unique Object Id of the AzureAD security group.

- Type <string>

  The type of AzureAD security group.

Notes

In the case of failure the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

- AzureADTenantIdMismatchAzureADAccessToken

  Mismatch between the given AzureADTenantID and TenantID in Azure AD AccessToken.

Related Links

- about_AcctADIdentitySnapin

Get-AcctDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the ADIdentity Service.

Syntax

```
Get-AcctDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected ADIdentity Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the ADIdentity Service instance running on controller "controller1.mydomain.net".

```
Get-AcctDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current ADIdentity Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the ADIdentityService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Set-AcctDBConnection
- Get-AcctServiceStatus
- Test-AcctDBConnection

Get-AcctDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix ADIdentity Service.

Syntax

```
Get-AcctDBSchema
   [-DatabaseName <String>]
   [-ServiceGroupName <String>]
   [-ScriptType <ScriptTypes>]
   [-LocalDatabase]
   [-Sid <String>]
   [-DatabaseRights <String>]
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets SQL scripts that can be used to create a new Citrix ADIdentity Service database schema, add a new ADIdentity service to an existing site, remove a ADIdentity service from a site, or create a database server logon for a ADIdentity service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected ADIdentity service instance, otherwise the scripts relate to ADIdentity service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to ADIdentity service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to ADIdentity service roles

If ScriptType is Evict, the returned script contains:

- Removal of ADIdentity service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix ADIdentity Service and copies it to a file called "C:\ADIdentitySchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a ADIdentity service schema.

```
Get-AcctDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\ADIdentitySchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the ADIdentity service. This can be used when configuring a mirror server for use.

```
Get-AcctDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    ADIdentityLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new ADIdentity service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the ADIdentity services that share the same database instance and are con-

sidered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix ADIdentity Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a ADIdentity Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a ADIdentity Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the ADIdentity Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for ADIdentity services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the ADIdentity Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Set-AcctDBConnection
- Test-AcctDBConnection

Get-AcctDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix ADIdentity Service.

Syntax

```
Get-AcctDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix ADIdentity Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix ADIdentity Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix ADIdentity Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix ADIdentity Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix ADIdentity Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-AcctServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-AcctDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix ADIdentity Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-AcctDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix ADIdentity Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any ADIdentity services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AcctServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctInstalledDBVersion
- Get-AcctServiceStatus
- Get-AcctDBSchema

Get-AcctIdentityPool

March 11, 2024

Gets a list of existing identity pools.

Syntax

```
Get-AcctIdentityPool
   [[-IdentityPoolName] <String>]
   [-IdentityPoolUid <Guid>]
   [-Lock <Boolean>]
   [-ScopeId <Guid>]
   [-ScopeName <String>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to retrieve a list of existing identity pools.

Examples

EXAMPLE 1

Gets all the identity pools.

```
Get-AcctIdentityPool

AvailableAccounts    : 1
DeviceManagementType : None
Domain               : MyDomain.com
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : MyPool1
IdentityPoolUid      : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
IdentityType         : ActiveDirectory
InUseAccounts        : 2
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : Acc####
NamingSchemeType     : Numeric
OU                   :
ResourceLocationId   : 59e9d28d-f46b-434a-97c2-995da9e120d7
StartCount           : 4
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
Scopes               :
TenantId             :

AvailableAccounts    : 0
DeviceManagementType : None
Domain               : MyDomain.com
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : Pool2
IdentityPoolUid      : 03743136-e43b-4a87-af74-ab71686b3c16
IdentityType         : ActiveDirectory
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : Test####
NamingSchemeType     : Alphabetic
OU                   :
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
```



```
Scopes                :
TenantId              :
```

EXAMPLE 2

Gets all the identity pools beginning with the character 'M'.

```
Get-AcctIdentityPool -IdentityPoolName M*

AvailableAccounts    : 0
DeviceManagementType : None
Domain               : MyDomain.com
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : MyPool1
IdentityPoolUid      : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
IdentityType         : ActiveDirectory
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : Acc####
NamingSchemeType     : Numeric
OU                   :
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
Scopes               :
TenantId             :
```

Parameters

-IdentityPoolName

The name of the identity pool.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-IdentityPoolUid

The unique identifier for the identity pool.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Lock

Whether the identity pool is locked or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets only results with a scope matching the specified scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets only results with a scope matching the specified scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	250

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Acct_Filtering for details.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

- AvailableAccounts <int>

  The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse', 'Tainted', or 'Error').

- DeviceManagementType <string>

  The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

- Domain <string>

  The Active Directory domain (in FQDN format) that accounts in the pool belong to.

- ErrorAccounts <int>

  The number of existing AD accounts in the 'Error'state.

- IdentityContent <string>

  JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

- IdentityPoolName <string>

  The name of the identity pool.

- IdentityPoolUid <GUID>

  The unique identifier for the identity pool.

- IdentityType <string>

  The identity type.

- InUseAccounts <int>

  The number of existing AD accounts in the 'InUse'state.

- Lock <bool>

  Indicates if the identity pool is locked.

- MetadataMap <IDictionary[string, string];>

  The metadata associated with this identity pool arranged in key value pairs.

- NamingScheme <string>

  The naming scheme for the identity pool.

- NamingSchemeType <string>

  The naming scheme type for the identity pool. This can be one of the following:

  - Numeric

    Naming scheme uses numeric indexes.

  - Alphabetic

    Naming scheme uses alphabetic indexes.

- OU <string>

  The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

- ResourceLocationId <GUID>

  The UID that corresponds to the resource location (DaaS only).

- StartCount <int>

  The next index to be used when creating an account in the identity pool.

- TaintedAccounts <int>

  The number of existing AD accounts in the 'Tainted'state.

- WorkgroupMachine <bool>

  If this is true, the identity pool can have an IdentityType of 'AzureAD'or 'Workgroup'.
  If this is false, the identity pool can have an IdentityType of 'ActiveDirectory'or 'HybridAzureAD'.

- ZoneUid <GUID>

  The UID that corresponds to the Zone in which AD accounts are created.

- Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

  The administration scopes associated with this identity pool.

- TenantId <GUID>

  Identity of the Citrix tenant associated with this identity pool.
  Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- CouldNotQueryDatabase

  The query required to get the database was not defined.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Remove-AcctIdentityPool
- Rename-AcctIdentityPool
- Set-AcctIdentityPool

Get-AcctInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the ADIdentity Service.

Syntax

```
Get-AcctInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix ADIdentity Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-AcctDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix ADIdentity Service database schema version number.

```
Get-AcctInstalledDBVersion
```

EXAMPLE 2

Get the versions of the ADIdentity Service database schema for which upgrade scripts are supplied.

```
Get-AcctInstalledDBVersion -Upgrade
```


Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-AcctInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the ADIdentity

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctDBVersionChangeScript
- Get-AcctDBSchema

Get-AcctScopedObject

March 11, 2024

Gets the details of the scoped objects for the ADIdentity Service.

Syntax

```
Get-AcctScopedObject
   [-ScopeId <Guid>]
   [-ScopeName <String>]
   [-ObjectType <ScopedObjectType>]
   [-ObjectId <String>]
   [-ObjectName <String>]
   [-Description <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Examples

EXAMPLE 1

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.

```
Get-AcctScopedObject -ObjectType Scheme

ScopeId     : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName   : Sales
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName   : Finance
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     :
ScopeName   :
ObjectType  : Scheme
ObjectId    : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
ObjectName  : AnotherScheme
Description : Another scheme in no scopes
```

Parameters

-ScopeId

Gets scoped object entries for the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ScopeName

Gets scoped object entries with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectType

Gets scoped object entries for objects of the given type.

Type:	ScopedObjectType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ObjectId

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectName

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Gets scoped object entries for objects with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.ScopedObject

The Get-AcctScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Specifies the unique identifier of the scope.

ScopeName <String>

Specifies the display name of the scope.

ObjectType <ScopedObjectType>

Type of the object this entry relates to.

ObjectId <String>

Unique identifier of the object.

ObjectName <String>

Display name of the object

Description <String>

Description of the object (possibly $null if the object type does not have a description).

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin

Get-AcctService

March 11, 2024

Gets the service record entries for the ADIdentity Service.

Syntax


```
Get-AcctService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the ADIdentity Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the ADIdentity Service running in the current service group.

```
Get-AcctService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```


Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*"matches records with a metadata entry having a key name of "abc"and a value starting with the letter "x".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.Service

The Get-AcctServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin

Get-AcctServiceAccount

March 11, 2024

Gets a list of existing service accounts.

Syntax

```
Get-AcctServiceAccount
    [-ServiceAccountUid <Guid>]
    [-IdentityProviderUid <Guid>]
    [-IsHealthy <Boolean>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to retrieve a list of existing service accounts.

Examples

EXAMPLE 1

Gets all the service accounts.

```
Get-AcctServiceAccount

ServiceAccountUid             : 440b77c3-9935-49b2-a696-01914dcffcbe
IdentityProviderIdentifier    : d1b46f6b-ad4c-4fcc-bb79-8e3c8fd813e8
IdentityProviderType          : AzureAD
SecretExpiryTime              : 9/8/2029 8:00:00 PM
AccountId                     : deb0811e-4839-4cce-87d3-8f36b31c2934
Capabilities                  : {
 AzureArcResourceManagement }

FailureReason                 :
IsHealthy                     : True
Scopes                        : {
 Scope1 }

TenantId                      :

ServiceAccountUid             : 2eccb0ec-a760-44ad-a36d-e0928b4d7926
IdentityProviderIdentifier    : d1b46f6b-ad4c-4fcc-bb79-8e3c8fd813e8
IdentityProviderType          : AzureAD
SecretExpiryTime              : 9/8/2099 8:00:00 PM
AccountId                     : ac14e785-cdb2-4e18-9240-8b49583b11a2
```



```
Capabilities                  : {
 AzureADDeviceManagement }

FailureReason                 :
IsHealthy                     : True
Scopes                        :
TenantId                      :
```

EXAMPLE 2

Gets the service account specified by uid 2eccb0ec-a760-44ad-a36d-e0928b4d7926.

```
Get-AcctServiceAccount -ServiceAccountUid 2eccb0ec-a760-44ad-a36d-
    e0928b4d7926

ServiceAccountUid             : 2eccb0ec-a760-44ad-a36d-e0928b4d7926
IdentityProviderIdentifier    : d1b46f6b-ad4c-4fcc-bb79-8e3c8fd813e8
IdentityProviderType          : AzureAD
SecretExpiryTime              : 9/8/2099 8:00:00 PM
AccountId                     : ac14e785-cdb2-4e18-9240-8b49583b11a2
Capabilities                  : {
 AzureADDeviceManagement }

FailureReason                 :
IsHealthy                     : True
Scopes                        :
TenantId                      :
```

Parameters

-ServiceAccountUid

The unique identifier for the service account.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IdentityProviderUid

The unique identifier for the identity provider that associates with the service account.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsHealthy

Whether the service account is healthy or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets only results with a scope matching the specified scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets only results with a scope matching the specified scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Acct_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Acct_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Acct_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.ServiceAccount

This object provides details of the service account and contains the following information:

- ServiceAccountUid <GUID>

  The unique identifier of the service account.

- SecretExpiryTime <Datetime>

  The expiration time for the secret of the service account.

- AccountId <string>

  The identifier for the service account. E.g. Azure application ID if the service account is with Azure AD as identity provider.

- IdentityProviderIdentifier <string>

  The identifier of the identity provider that the service account belongs to. E.g. Azure AD tenant ID.

- IdentityProviderType <string>

  The type of the identity provider of the service account. Can be AzureAD or ActiveDirectory.

- IsHealthy <bool>

  Indicates if the service account is healthy.

- Capabilities <string[]>

  Capabilities of the service account. Can be AzureArcResourceManagement.

- FailureReason <string>

  The reason why the service account becomes unhealthy.

- Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

  The administration scopes associated with this identity pool.

- TenantId <GUID>

  Identity of the Citrix tenant associated with this identity pool.
  Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure the following errors can result:

- PartialData

  Only a subset of the available data was returned.

- CouldNotQueryDatabase

  The query required to get the database was not defined.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- CommunicationError

  An error occurred while communicating with the service.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- InvalidFilter

  A filtering expression was supplied that could not be interpreted for this cmdlet.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctServiceAccount
- Remove-AcctServiceAccount
- Set-AcctServiceAccount

Get-AcctServiceAddedCapability

March 11, 2024

Gets any added capabilities for the ADIdentity Service on the controller.

Syntax

```
Get-AcctServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the ADIdentity Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the ADIdentity Service.

```
Get-AcctServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin

Get-AcctServiceInstance

March 11, 2024

Gets the service instance entries for the ADIdentity Service.

Syntax

```
Get-AcctServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the ADIdentity Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the ADIdentity Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-AcctServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.ServiceInstance

The Get-AcctServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Acct.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.ADIdentity.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctServiceStatus
- Reset-AcctServiceGroupMembership

Get-AcctServiceStatus

March 11, 2024

Gets the current state of the ADIdentity Service on the controller.

Syntax

```
Get-AcctServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the ADIdentity Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the ADIdentity Service.

```
1 Get-AcctServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-AcctServiceStatus command returns an object containing the status of the ADIdentity Service together with extra diagnostics information.

DBUnconfigured

The ADIdentity Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the ADIdentity Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the ADIdentity Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The ADIdentity is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The ADIdentity is connected to a database that is valid, but it does not have the full functionality required so the ADIdentity cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the ADIdentity Service currently in use is incompatible with the version of the ADIdentity Service schema on the database. Upgrade the ADIdentity Service to a more recent version.

DBOlderVersionThanService

The version of the ADIdentity Service schema on the database is incompatible with the version of the ADIdentity Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The ADIdentity Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the ADIdentity Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the ADIdentity and the database has been lost for an extended period of time, or has failed due to a configuration problem. The ADIdentity service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Set-AcctDBConnection
- Test-AcctDBConnection
- Get-AcctDBConnection
- Get-AcctDBSchema

Import-AcctIdentityPool

March 11, 2024

Imports an identity pool in the site

Syntax

```
Import-AcctIdentityPool
      -IdentityPoolData <String>
      [-Scope <String[]>]
      [-ZoneUid <Guid>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to import an identity pool from another site that has been exported using Export-AcctIdentityPool. Any AD accounts in the exported identity pool will be imported as well.

Examples

EXAMPLE 1

Import the identity pool defined in the variable $identityPoolData. This variable should be created using Export-AcctIdentityPool (potentially on a different site). The imported identity pool contains any accounts that were exported with the identity pool.

```
Import-AcctIdentityPool -IdentityPoolData $identityPoolData
        c:\PS> Get-AcctIdentityPool -IdentityPoolName MyPool

          AvailableAccounts    : 0
          DeviceManagementType : None
          Domain               : MyDomain.com
          ErrorAccounts        : 0
          IdentityContent      :
          IdentityPoolName     : MyPool
          IdentityPoolUid      : 5a2ac1c4-b183-4796-a984-163b645c6ca4
          IdentityType         : ActiveDirectory
          InUseAccounts        : 2
          Lock                 : False
          MetadataMap          : {
  }
```



```
16
17              NamingScheme          : Acc###
18              NamingSchemeType      : Numeric
19              OU                    :
20              ResourceLocationId    :
21              StartCount            : 3
22              TaintedAccounts       : 0
23              WorkgroupMachine      : False
24              ZoneUid               :
25              Scopes                :
26              TenantId              :
```

Parameters

-IdentityPoolData

Specifies the identity pool to import in the form of a JSON encoded string.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Scope

The administration scopes to be applied to the new identity pool.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

The UID that corresponds to the Zone in which these AD accounts will be created. This is only intended to be used for Citrix Cloud Delivery Controllers.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

- AvailableAccounts <int>

  The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse', 'Tainted', or 'Error').

- DeviceManagementType <string>

  The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

- Domain <string>

  The Active Directory domain (in FQDN format) that accounts in the pool belong to.

- ErrorAccounts <int>

  The number of existing AD accounts in the 'Error'state.

- IdentityContent <string>

  JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

- IdentityPoolName <string>

  The name of the identity pool.

- IdentityPoolUid <GUID>

  The unique identifier for the identity pool.

- IdentityType <string>

  The identity type.

- InUseAccounts <int>

  The number of existing AD accounts in the 'InUse'state.

- Lock <bool>

  Indicates if the identity pool is locked.

- MetadataMap <IDictionary[string, string];>

  The metadata associated with this identity pool arranged in key value pairs.

- NamingScheme <string>

  The naming scheme for the identity pool.

- NamingSchemeType <string>

  The naming scheme type for the identity pool. This can be one of the following:

  - Numeric

    Naming scheme uses numeric indexes.

  - Alphabetic

    Naming scheme uses alphabetic indexes.

- OU <string>

  The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

- ResourceLocationId <GUID>

  The UID that corresponds to the resource location (DaaS only).

- StartCount <int>

  The next index to be used when creating an account in the identity pool.

- TaintedAccounts <int>

  The number of existing AD accounts in the 'Tainted'state.

- WorkgroupMachine <bool>

  If this is true, the identity pool can have an IdentityType of 'AzureAD'or 'Workgroup'.
  If this is false, the identity pool can have an IdentityType of 'ActiveDirectory'or 'HybridAzureAD'.

- ZoneUid <GUID>

  The UID that corresponds to the Zone in which AD accounts are created.

- Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

  The administration scopes associated with this identity pool.

- TenantId <GUID>

  Identity of the Citrix tenant associated with this identity pool.
  Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure, the following errors can result.

Error Codes

CouldNotQueryDatabase

The query required to get the database was not defined.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Export-AcctIdentityPool
- Export-ProvScheme
- Import-ProvScheme

New-AcctADAccount

March 11, 2024

Creates Active Directory (AD) computer accounts in the specified identity pool.

Syntax

```
New-AcctADAccount
   [-IdentityPoolName] <String>
   -Count <Int32>
   [-StartCount <Int32>]
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Parallelism <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AcctADAccount
   [-IdentityPoolName] <String>
   -ADAccountName <String[]>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Parallelism <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AcctADAccount
   -IdentityPoolUid <Guid>
   -Count <Int32>
   [-StartCount <Int32>]
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Parallelism <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AcctADAccount
   -IdentityPoolUid <Guid>
   -ADAccountName <String[]>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Parallelism <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Provides the ability to create new Active Directory (AD) computer accounts and register them in an already existing identity pool.

The accounts are created using the information stored in the identity pool. This provides the account name (via the Naming Scheme property and Start Count), domain, and OU.

The user performing the operation in PowerShell must have sufficient privileges in AD to create the new computer accounts. If the current user does not have sufficient privileges, the ADUserName and ADPassword parameters may be used instead.

The AD account names will pad the index to use all the space specified in the identity pool naming scheme (e.g. "acc###"will become "acc001"). However, if the index overflows the available space the cmdlet expands the format to use the next incremental number (e.g. "acc###"will become "acc1000" if the index is 10000, which cannot fit into the three '#'placeholders). If this expanded name exceeds the 15 character name limit, the accounts are not created.

There can be only one creation process running for a specific identity pool at any one time. Attempting to start another account creation process while an existing one is executing results in an error being returned.

Examples

EXAMPLE 1

Creates two new AD accounts and registers them in the identity pool called "MyPool".

```
New-AcctADAccount -IdentityPoolName MyPool -Count 2 -OutVariable result

SuccessfulAccounts                SuccessfulAccountsCount
   FailedAccountsCount    FailedAccounts
------------------                -----------------------
   -------------------    --------------
{
 MyDomain\ACC001, MyDomain\ACC002 }
  2                              0                      {
  }


$result[0].SuccessfulAccounts

ADAccountGuid          : a33f54f8-4944-4537-93c9-a04f0b889378
ADAccountName          : MyDomain\ACC001
ADAccountSid           : S-1-5-21-1315084875-1285793635-2418178940-2684
AccountDisabled        : False
AccountLocked          : False
Domain                 : MyDomain.com
```



```
DomainControllerHint   :
    v2_ZGMubXlkb21haW4uY29tOjU5ZTlkMjhkLWY0NmItNDM0YS05N2MyLTk5NWRhOWUxMjBkNw
    ==
Lock                   : False
State                  : Available
TenantId               :
DeviceManagementType   : None
IdentityType           : ActiveDirectory
VdaHostId              : ee3ec984-3f1b-41ed-aee7-38754692e829
WorkgroupMachine       : False
TrustServiceInstanceId : ee3ec984-3f1b-41ed-aee7-38754692e829-S
    -1-5-21-1315084875-1285793635-2418178940-2684

ADAccountGuid          : 16c62c01-42d0-4180-91c3-3e02b5a3e860
ADAccountName          : MyDomain\ACC002
ADAccountSid           : S-1-5-21-1315084875-1285793635-2418178940-2685
AccountDisabled        : False
AccountLocked          : False
Domain                 : MyDomain.com
DomainControllerHint   :
    v2_ZGMubXlkb21haW4uY29tOjU5ZTlkMjhkLWY0NmItNDM0YS05N2MyLTk5NWRhOWUxMjBkNw
    ==
Lock                   : False
State                  : Available
TenantId               :
DeviceManagementType   : None
IdentityType           : ActiveDirectory
VdaHostId              : 8ca4ee0c-00bf-4e05-8605-c3ee7fe053c3
WorkgroupMachine       : False
TrustServiceInstanceId : 8ca4ee0c-00bf-4e05-8605-c3ee7fe053c3-S
    -1-5-21-1315084875-1285793635-2418178940-2685
```

EXAMPLE 2

Creates two new AD accounts and registers them in the identity pool called “MyPool”, starting from an index of 50.

```
New-AcctADAccount -IdentityPoolName MyPool -Count 2 -StartCount 50 -
    OutVariable result

SuccessfulAccounts                SuccessfulAccountsCount
    FailedAccountsCount    FailedAccounts
------------------                -----------------------
    --------------------    --------------
{
 MyDomain\ACC050, MyDomain\ACC051 }
  2                             0                       {
  }


$result[0].SuccessfulAccounts
```



```
12
13 ADAccountGuid          : fed20f3b-d1ec-4f8e-9681-197de9522bfd
14 ADAccountName          : MyDomain\ACC050
15 ADAccountSid           : S-1-5-21-1315084875-1285793635-2418178940-2686
16 AccountDisabled        : False
17 AccountLocked          : False
18 Domain                 : MyDomain.com
19 DomainControllerHint   :
      v2_ZGMubXlkb21haW4uY29tOjU5ZTlkMjhkLWY0NmItNDM0YS05N2MyLTk5NWRhOWUxMjBkNw
      ==
20 Lock                   : False
21 State                  : Available
22 TenantId               :
23 DeviceManagementType   : None
24 IdentityType           : ActiveDirectory
25 VdaHostId              : 1772a527-33e6-4da4-8a3a-d98e13c1d156
26 WorkgroupMachine       : False
27 TrustServiceInstanceId : 1772a527-33e6-4da4-8a3a-d98e13c1d156-S
      -1-5-21-1315084875-1285793635-2418178940-2686
28
29 ADAccountGuid          : 86b3b360-e21f-4bb2-8e36-e236598fd51a
30 ADAccountName          : MyDomain\ACC051
31 ADAccountSid           : S-1-5-21-1315084875-1285793635-2418178940-2687
32 AccountDisabled        : False
33 AccountLocked          : False
34 Domain                 : MyDomain.com
35 DomainControllerHint   :
      v2_ZGMubXlkb21haW4uY29tOjU5ZTlkMjhkLWY0NmItNDM0YS05N2MyLTk5NWRhOWUxMjBkNw
      ==
36 Lock                   : False
37 State                  : Available
38 TenantId               :
39 DeviceManagementType   : None
40 IdentityType           : ActiveDirectory
41 VdaHostId              : 0185a831-493f-4656-9126-9a29188c8213
42 WorkgroupMachine       : False
43 TrustServiceInstanceId : 0185a831-493f-4656-9126-9a29188c8213-S
      -1-5-21-1315084875-1285793635-2418178940-2687
```

Parameters

-IdentityPoolName

The name of the identity pool in which accounts will be created.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Count

The number of accounts to create.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountName

The AD account names to be created. These are just the simple machine account names (e.g. MyVM001).

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool in which accounts will be created.

Type:	Guid

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StartCount

The start index for the create process.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADUserName

The username for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADPassword

The matching password for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Parallelism

Gets or sets the maximum number of threads to create AD accounts in parallel. Range 1-60 with default of 20.

Type:	Int32
Position:	Named
Default value:	20
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationDetailedSummary

The New-AcctADAccount returns an object that contains the following properties:

- SuccessfulAccountsCount <int>

  The number of accounts that were added successfully

- FailedAccountsCount <int>

  The number of accounts that were not added.

- FailedAccounts <Citrix.XDPowerShell.AccountError[]>

  The list of accounts that failed to be added. Each one has the following properties:

  - ADAccountName <string>
  - ADAccountSid <String>
  - ErrorReason <ADIdentityStatus>

    This can be one of the following:

    * IdentityDuplicateObjectExists

      An identity with the same SID already exists.

* ADServiceDatabaseError

  An error occurred in the service while attempting a database operation.

* ADServiceDatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

* ADServiceStatusInvalidDb

  An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

* FailedToConnectToDomainController

  Contacting Active Directory failed.

* FailedToGetOrganizationUnitInAD

  Failed to access the OU in Active Directory.

* FailedToGetDefaultComputerContainerInAD

  Failed to access the default computers container in Active Directory.

* FailedToCreateComputerAccountInAD

  Failed to create the computer account in Active Directory.

* FailedToAccessComputerAccountInAD

  Failed to read the newly created computer account in Active Directory.

* FailedToGetSidFromAD

  Failed to get the SID for the created account from Active Directory.

* FailedToSetSamAccountNameInAD

  Failed to set the SAM account name in Active Directory for the account created.

* FailedToSetUserAccountControlInAD

  Failed to set the user account controller properties for the account created in Active Directory.

* FailedToSaveChangeInAD

  Failed to save the changes made to the created computer account in Active Directory.

* FailedToSetPasswordInAD

  Failed to set the password for the created computer account in Active Directory.

        * FailedToEnableAccountInAD

          Failed to enable the newly created computer account in Active Directory.

        * ComputerNameAlreadyInUseInAD

          The computer name for the computer to create is in use in Active Directory.

        * FailedToGetDistinguishedNameInAD

          Failed to get the distinguished name for the created computer account in ActiveDirectory.

        * FailedToSetDnsHostNameInAD

          Failed to set the Dns Host Name property for the created computer account in ActiveDirectory.

        * FailedToSetDisplayNameInAD

          Failed to set the DisplayName property for the created computer account in ActiveDirectory.

        * FailedToWriteServicePrincipalNameInAD

          Failed to set the ServicePrincipalName property for the created computer account in ActiveDirectory.

    - DiagnosticInformtion <Exception>

      Any other error information

- SuccessfulAccounts <Citrix.ADIdentity.Sdk.Identity[]>

  The list of accounts that were successfully added. Each one has the following properties:

    - ADAccountGuid <Guid>

      The unique identifier for the account.

    - ADAccountName <string>

      The name of the account.

    - ADAccountSid <string>

      The SID for the account.

    - AccountDisabled <bool>

      Whether or not the account is disabled in AD.

    - AccountLocked <bool>

      Whether or not the account is locked in AD.

- Domain <string>

  The domain for the account.

- DomainControllerHint <string>
- Lock <bool>

  Whether or not the account is locked (in the database, not AD).

- State <Citrix.ADIdentity.Sdk.ADIdentityState>

  The state for the account. This can be:

  * Available

    The account is not used.

  * InUse

    The account is in use.

  * Error

    The account is in error (i.e. the account is locked or disabled in AD).

  * Tainted

    The account is no longer used, but the password is no longer known.

- TenantId <Guid>

  The identity of the tenant associated with this account.

- DeviceManagementType <string>

  The device management type.

- IdentityType <string>

  The identity type.

- VdaHostId <Guid>

  The ID of the VDA associated with this account.

- WorkgroupMachine <bool>

  Whether or not the account is a workgroup account (not domain-joined).

- TrustServiceInstanceId <string>

  The trust service ID of the machine.

Notes

In the case of failure the following errors can result:

- NamingSchemeNotSpecifiedForIdentityPool

  No naming scheme is defined in the specified identity pool.

- IdentityPoolObjectNotFound

  The specified identity pool was not located.

- IdentityPoolAlreadyLocked

  The specified identity pool is locked.

- PermissionDenied

  The user does not have administrative rights to perform this operation.

- ConfigurationLoggingError

  The operation could not be performed because of a configuration logging error

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- ServiceStatusInvalidDb

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- CommunicationError

  An error occurred while communicating with the service.

- ExceptionThrown

  An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Add-AcctADAccount
- Remove-AcctADAccount
- Get-AcctADAccount

- Repair-AcctADAccount
- Unlock-AcctADAccount
- Update-AcctADAccount

New-AcctIdentityPool

March 11, 2024

Creates a new identity pool.

Syntax

```
New-AcctIdentityPool
   -IdentityPoolName <String>
   [-NamingScheme <String>]
   [-NamingSchemeType <ADIdentityNamingScheme>]
   [-OU <String>]
   [-Domain <String>]
   [-AllowUnicode]
   [-IdentityType <String>]
   [-DeviceManagementType <String>]
   [-StartCount <Int32>]
   [-Scope <String[]>]
   [-TenantId <Guid>]
   [-ZoneUid <Guid>]
   [-AzureADSecurityGroupName <String>]
   [-AzureADAccessToken <String>]
   [-AzureADTenantId <Guid>]
   [-ServiceAccountUid <Guid[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AcctIdentityPool
   -IdentityPoolName <String>
   [-NamingScheme <String>]
   [-NamingSchemeType <ADIdentityNamingScheme>]
   [-AllowUnicode]
   [-WorkgroupMachine]
   [-IdentityType <String>]
   [-DeviceManagementType <String>]
   [-StartCount <Int32>]
   [-Scope <String[]>]
   [-TenantId <Guid>]
   [-ZoneUid <Guid>]
   [-AzureADSecurityGroupName <String>]
   [-AzureADAccessToken <String>]
```



```
    [-AzureADTenantId <Guid>]
    [-ServiceAccountUid <Guid[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to create identity pools that can be used to store AD computer accounts.

The naming scheme, naming scheme type, and domain must be specified if the identity pool is to be used to create new accounts.

Each identity pool is tied to a single domain. All the identities in an identity pool belong to the same domain. If the domain is not specified by a parameter to this command the domain will be set when an account is imported into it.

Examples

EXAMPLE 1

Create a new identity pool from which accounts can be created in the domain called "MyDomain.com" using a numeric scheme of the format Acc####. The first account created from this identity pool will be named Acc0001 and placed in the OU named "MyOU".

New AD accounts can be imported into this pool too, but must be from the Domain "MyDomain.com" .

```
New-AcctIdentityPool -IdentityPoolName MyPool -NamingScheme Acc#### -
    Domain MyDomain.com -NamingSchemeType Numeric -OU "CN=MyOU,DC=
    MyDomain,DC=com"


AvailableAccounts    : 0
DeviceManagementType : None
Domain               : MyDomain.com
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : MyPool
IdentityPoolUid      : 22072d9e-6a8f-494b-a5bc-2ef18ca4b915
IdentityType         : ActiveDirectory
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : Acc####
```



```
NamingSchemeType     : Numeric
OU                   : CN=MyOU,DC=MyDomain,DC=com
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
Scopes               :
TenantId             :
```

EXAMPLE 2

Create a new identity pool named “MyWorkgroupPool”where accounts are created as workgroup machines rather than part of a domain. Note that the OU and Domain parameters must be unused.

```
New-AcctIdentityPool -WorkgroupMachine -IdentityPoolName
    MyWorkgroupPool -NamingScheme Acc####  -NamingSchemeType Numeric


AvailableAccounts    : 0
DeviceManagementType : None
Domain               :
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : MyWorkgroupPool
IdentityPoolUid      : f4aef7af-4298-44a3-a5fb-4a9201ca01d7
IdentityType         : Workgroup
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : Acc####
NamingSchemeType     : Numeric
OU                   :
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : True
ZoneUid              :
Scopes               : {
  }

TenantId             :
```

EXAMPLE 3

Create a new identity pool named “AzureADIdentityPool”using the AzureAD identity type. Accounts created in this pool are joined to Azure Active Directory, and Microsoft Intune is used for device man-

agement.

```
New-AcctIdentityPool -IdentityPoolName AzureADIdentityPool -
    DeviceManagementType Intune -IdentityType AzureAD -NamingScheme
    AzureAD-### -NamingSchemeType Numeric -WorkgroupMachine

AvailableAccounts    : 0
DeviceManagementType : Intune
Domain               :
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : AzureADIdentityPool
IdentityPoolUid      : a0208d3f-7467-4cec-b6cb-b3e14560e1e7
IdentityType         : AzureAD
InUseAccounts        : 0
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : AzureAD-###
NamingSchemeType     : Numeric
OU                   :
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : True
ZoneUid              :
Scopes               : {
  }

TenantId             :
```

EXAMPLE 4

Create a new identity pool named “HybridAzureADIdentityPool”using the HybridAzureAD identity type. Accounts created in this pool are joined to both the domain “IZONE.CLOUD”and Azure Active Directory.

```
New-AcctIdentityPool -IdentityPoolName HybridAzureADIdentityPool -
    IdentityType HybridAzureAD -NamingScheme HAAD-### -NamingSchemeType
    Numeric -Domain izone.cloud -OU "CN=Computers,DC=izone,DC=cloud"

AvailableAccounts    : 0
DeviceManagementType : None
Domain               : IZONE.CLOUD
ErrorAccounts        : 0
IdentityContent      :
IdentityPoolName     : HybridAzureADIdentityPool
IdentityPoolUid      : d08ffd26-6600-46d9-b649-786f201082b3
IdentityType         : HybridAzureAD
InUseAccounts        : 0
```



```
Lock                 : False
MetadataMap          : {
  }

NamingScheme         : HAAD-###
NamingSchemeType     : Numeric
OU                   : CN=Computers,DC=izone,DC=cloud
ResourceLocationId   :
StartCount           : 1
TaintedAccounts      : 0
WorkgroupMachine     : False
ZoneUid              :
Scopes               : {
  }

TenantId             :
```

Parameters

-IdentityPoolName

The name of the identity pool. This must not contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-WorkgroupMachine

Indicates whether the accounts created should be part of a workgroup rather than a domain.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-NamingScheme

Defines the template name for AD accounts created in the identity pool. The scheme can consist of fixed characters and a variable part defined by '#' characters. There can be only one variable region defined. The number of '#' characters defines the minimum length of the variable region. For example, a naming scheme of H#### could create accounts called H0001, H0002 (for a numeric scheme type) or HAAAA, HAAAB (for an alphabetic type).

Citrix recommends that you define a naming scheme that will not clash with accounts which already exist in AD; account creation will fail if a clash occurs.

There are restrictions on what constitutes a valid computer name:

Minimum name length: 2 (DNS)

Maximum name length: 15 bytes (NetBIOS)

The following characters are not allowed in a computer name:

backslash (\)

slash mark (/)

colon (:)

asterisk (*)

question mark (?)

quotation mark (")

less than sign (<)

greater than sign (>)

vertical bar (|)

comma (,)

tilde (~)

exclamation point (!)

at sign (@)

number sign (#)

dollar sign ($)

percent (%)

caret (^)

ampersand (&)

apostrophe (‘)

parenthesis (())

braces ({})

underscore (_)

The following names are reserved and must not be used at the end of the naming scheme:

-GATEWAY

-GW

-TAC

Names must not start with a period (NetBIOS).

Names must not be composed entirely of numbers (NetBIOS).

Names must not contain a blank or space characters (DNS).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NamingSchemeType

The type of naming scheme. This can be Numeric or Alphabetic. This defines the format of the variable part of the AD account names that will be created.

Type:	ADIdentityNamingScheme
Position:	Named
Default value:	Numeric
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AllowUnicode

Allow the naming scheme to have characters other than alphanumeric characters.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityType

The type of identity type. This can be ActiveDirectory, AzureAD, HybridAzureAD, or Workgroup.

Type:	String
Accepted values:	ActiveDirectory, AzureAD, HybridAzureAD, Workgroup
Position:	Named
Default value:	ActiveDirectory
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeviceManagementType

The type of device management type. This can be Intune, IntuneWithCitrixTags, or None.

Type:	String

Accepted values:	Intune, IntuneWithCitrixTags, None
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartCount

Specifies the next number to be used when creating new AD accounts in the identity pool.

Type:	Int32
Position:	Named
Default value:	1
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Scope

The administration scopes to be applied to the new identity pool.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with identity pool. Must always be specified in multitenant sites, must not be specified otherwise.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

The UID that corresponds to the Zone in which these AD accounts will be created. This is only intended to be used for Citrix Cloud Delivery Controllers.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADSecurityGroupName

The name of AzureAD security group. This is only intended to be used when IdentityType is “AzureAD” or “HybridAzureAD”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADAccessToken

The access token of Microsoft Graph API. Must be specified if “AzureADSecurityGroupName”is specified to create Azure AD security group, and IdentityType is “AzureAD”or “HybridAzureAD”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADTenantId

The tenantId of AzureAD. Must be specified if “AzureADSecurityGroupName”is specified to create Azure AD security group, and IdentityType is “AzureAD”or “HybridAzureAD”.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceAccountUid

Specify the UID of the service account that associates with the identity pool.

Type:	Guid[]

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OU

The OU that computer accounts will be created in. If this is not specified, accounts are created into the default account container specified by AD. This is the 'Computers'container for out-of-the-box installations of AD. The OU must be a valid AD container of the domain specified for the pool.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Domain

The AD domain name for the pool. Specify this in FQDN format; for example, MyDomain.com.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

AvailableAccounts <int>

The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse', 'Tainted', or 'Error').

DeviceManagementType <string>

The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

Domain <string>

The Active Directory domain (in FQDN format) that accounts in the pool belong to.

ErrorAccounts <int>

The number of existing AD accounts in the 'Error' state.

IdentityContent <string>

JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

IdentityPoolName <string>

The name of the identity pool.

IdentityPoolUid <GUID>

The unique identifier for the identity pool.

IdentityType <string>

The identity type.

InUseAccounts <int>

The number of existing AD accounts in the 'InUse' state.

Lock <bool>

Indicates if the identity pool is locked.

MetadataMap <IDictionary[string, string];>

The metadata associated with this identity pool arranged in key value pairs.

NamingScheme <string>

The naming scheme for the identity pool.

NamingSchemeType <string>

The naming scheme type for the identity pool. This can be one of the following:

Numeric - naming scheme uses numeric indexes

Alphabetic - naming scheme uses alphabetic indexes

OU <string>

The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

ResourceLocationId <GUID>

The UID that corresponds to the resource location (DaaS only).

StartCount <int>

The next index to be used when creating an account in the identity pool.

TaintedAccounts <int>

The number of existing AD accounts in the 'Tainted' state.

WorkgroupMachine <bool>

If this is true, the identity pool can have an IdentityType of 'AzureAD' or 'Workgroup'.

If this is false, the identity pool can have an IdentityType of 'ActiveDirectory' or 'HybridAzureAD'.

ZoneUid <GUID>

The UID that corresponds to the Zone in which AD accounts are created.

Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

The administration scopes associated with this identity pool.

TenantId <GUID>

Identity of the Citrix tenant associated with this identity pool.

Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure, the following errors can result.

Error Codes

UnableToConvertDomainName

Unable to convert domain name to DNS format.

NamingSchemeNotEnoughCharacters

Naming scheme does not have enough characters specified.

NamingSchemeTooManyCharacters

Naming scheme has too many characters specified.

NamingSchemeIllegalCharacter

Naming scheme contains illegal characters.

NamingSchemeMayNotStartWithPeriod

Naming scheme starts with a period (.) character.

NamingSchemeMayNotBeAllNumbers

Naming scheme contains only numbers.

NamingSchemeMissingNumericSpecifications

Naming scheme does not contain any variable specification (i.e. no '#'characters are specified).

NamingSchemeHasMoreThanOneSetOfHashes

Naming scheme has more than one variable region (i.e. there are '#'characters separated by other characters).

IdentityPoolDuplicateObjectExists

An identity pool with the same name already exists.

IdentityPoolOUInvalid

Identity pool OU invalid as it does not exist.

IdentityPoolOUOfWrongDomain

Identity pool OU invalid as it refers to a different domain to the domain specified for the pool.

InvalidIdentityPoolParameterCombination

Caused by either of the following validation errors:

- If an OU is specified then a domain must also be specified.
- NamingScheme, NamingSchemeType and Domain must all be present if any of these are specified.

OUNotReachable

Domain is not reachable. Check if the domain exists in any zones and if the domain contains the specified OU.

OUParameterRequiresDomain

If an OU is specified then a domain must also be specified.

NamingSchemeAndNamingSchemeTypeMustBeUsedTogether

NamingScheme and NamingSchemeType must be used together.

NamingSchemeIllegalComputerName

The naming scheme supplied is not valid.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

AzureADAccessTokenMustBeSpecified

Must specify “AzureADAccessToken”parameter to operate AzureAD security group.

AzureADSecurityGroupNotSupportCurrentIdentityType

Current IdentityType not support AzureAD security group.

AzureADSecurityGroupMustBeSpecified

Must specify AzureAD security group.

AzureADTenantIdMustBeSpecified

AzureAD TenantId must be specified.

AzureADTenantIdInIdentityContentMismatchAzureADAccessToken

The TenantId extracted from AccessToken must be the same as specified TenantId in IdentityContent.

AzureADTenantIdMismatchAzureADAccessToken

The given AzureADTenantId mismatchs TenantId contained in AzureAD AccessToken.

CurrentIdentityTypeRequiresWorkgroupMachine

Current IdentityType requires WorkgroupMachine.

DomainNotRequiredForAzureADAndWorkgroup

Domain not required for AzureAD and Workgroup.

FailedToCreateAzureADSecurityGroup

An error occured in the service while creating AzureAD security group.

FailedToDeleteAzureADSecurityGroup

An error occured in the service while deleting AzureAD security group.

NamingSchemeAndDomainMustBeUsedTogetherForActiveDirectoryOrHybridAzureAD

NamingScheme and Domain must be used together for ActiveDirectory or HybridAzureAD.

NewAzureADSecurityGroupWithoutFailureButNoGroupResultReturned

No result returned when create AzureAD security group though doesn't get error when execute cmdlet.

IntuneEnrollRequiresAzureADJoinedOrHybridAzureADJoined

Intune enroll requires AzureAD Joined or Hybrid AzureAD Joined.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctIdentityPool
- Rename-AcctIdentityPool
- Set-AcctIdentityPool
- Test-AcctIdentityPoolNameAvailable
- New-AcctADAccount

New-AcctServiceAccount

March 11, 2024

Creates a new service account.

Syntax


```
New-AcctServiceAccount
   -IdentityProviderType <String>
   -IdentityProviderIdentifier <String>
   -AccountId <String>
   -AccountSecret <SecureString>
   -SecretExpiryTime <DateTime>
   [-Capabilities <String[]>]
   [-Scope <String[]>]
   [-TenantId <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to create service accounts that can be used to store credential to access identity provider like Azure AD.

Each service account is tied to a single identity provider.

Examples

EXAMPLE 1

Create a new service account to access AzureAD with AzureArcResourceManagement capability.

```
New-AcctServiceAccount -IdentityProviderType AzureAD -
   IdentityProviderIdentifier f439f4c0-fcd8-4fe6-95b8-71e7e49dc8c6 -
   AccountId ac14e785-cdb2-4e18-9240-8b49583b11a2 -AccountSecret
   $password -SecretExpiryTime 2024-09-09 -Capabilities '
   AzureArcResourceManagement'

ServiceAccountUid             : 17631afc-2e4c-491e-b0aa-f979a80e32c1
IdentityProviderIdentifier    : f439f4c0-fcd8-4fe6-95b8-71e7e49dc8c6
IdentityProviderType          : AzureAD
SecretExpiryTime              : 9/9/2024 8:00:00 PM
AccountId                     : ac14e785-cdb2-4e18-9240-8b49583b11a2
Capabilities                  : {
 AzureArcResourceManagement }

FailureReason                 :
IsHealthy                     : True
Scopes                        : {
  }

TenantId                      :
```


EXAMPLE 2

Create a new service account to access ActiveDirectory.

```
New-AcctServiceAccount -IdentityProviderType ActiveDirectory -
    IdentityProviderIdentifier test.local -AccountId test\svcacct_1 -
    AccountSecret $password -SecretExpiryTime 2024-09-09

ServiceAccountUid             : ad24284e-ba3d-4504-80db-9ac6640de533
IdentityProviderIdentifier    : test.local
IdentityProviderType          : ActiveDirectory
SecretExpiryTime              : 9/9/2024 8:00:00 PM
AccountId                     : test\svcacct_1
Capabilities                  :
FailureReason                 :
IsHealthy                     : True
Scopes                        : {
  }

TenantId                      :
```

Parameters

-IdentityProviderType

The type of the identity provider that associates with this service account. Can be AzureAD or ActiveDirectory.

Type:	String
Accepted values:	ActiveDirectory, AzureAD
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityProviderIdentifier

The identifier of the given identity provider. E.g. Azure AD tenant id if 'IdentityProviderType'is AzureAD.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AccountId

The identifier for the service account. E.g. Azure application (client) id if 'IdentityProviderType'is AzureAD.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AccountSecret

The secret for the service account. E.g. Azure application (client) secret if 'IdentityProviderType'is AzureAD. The secret will be encrypted and stored in database.

Type:	SecureString
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SecretExpiryTime

The secret expiration time for the service account.

Type:	DateTime
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Capabilities

The capabilities for the service account.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Scope

The administration scopes to be applied to the new service account.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with service account. Must always be specified in multi-tenant sites, must not be specified otherwise.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.ServiceAccount

This object provides details of the service account and contains the following information:

ServiceAccountUid <GUID> The unique identifier of the service account. SecretExpiryTime <Datetime> The expiration time for the secret of the service account. AccountId <string> The identifier for the service account. E.g. Azure application ID if the service account is with Azure AD as identity provider. IdentityProviderIdentifier <string> The identifier of the identity provider that the service account belongs to. E.g. Azure AD tenant ID. IdentityProviderType <string> The type of the identity provider of the service account. Can be AzureAD or ActiveDirectory. IsHealthy <bool> Indicates if the service account is healthy. Capabilities <string[]> Capabilities of the service account. Can be AzureArcResourceManagement. FailureReason <string> The reason why the service account becomes unhealthy. Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]> The administration scopes associated with this identity pool. TenantId <GUID> Identity of the Citrix tenant associated with this identity pool. Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure, the following errors can result.

Error Codes

IdentityProviderTypeDoesNotMatch

The specified identity provider type doesn't match the existing identity provider for the given tenant.

ServiceAccountDuplicateObjectExists

Duplicate object exists.

UnsupportedIdentityProviderType

The specified identity provider type is not supported.

InvalidServiceAccountCapabilities

One or more specified service account capabilities are not supported.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctServiceAccount
- Set-AcctServiceAccount
- Get-AcctServiceAccount

Remove-AcctADAccount

March 11, 2024

Removes Active Directory (AD) computer accounts from an identity pool.

Syntax

```
Remove-AcctADAccount
      [-IdentityPoolName] <String>
      -ADAccountName <String[]>
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-RemovalOption <ADIdentityRemoveAccountOption>]
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctADAccount
      [-IdentityPoolName] <String>
      -ADAccountSid <String[]>
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-RemovalOption <ADIdentityRemoveAccountOption>]
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctADAccount
      -IdentityPoolUid <Guid>
      -ADAccountName <String[]>
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-RemovalOption <ADIdentityRemoveAccountOption>]
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctADAccount
      -IdentityPoolUid <Guid>
      -ADAccountSid <String[]>
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-RemovalOption <ADIdentityRemoveAccountOption>]
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove Active Directory (AD) accounts from an identity pool. This only removes the AD account from the identity pool within the AD Identity Service. Options for removing (or dis-

abling) the account in AD are available if necessary.

All aspects of this command that need to make modifications to the accounts in AD will use the account that the runspace is using. This means that if an account is to be removed from AD or disabled, the user performing the operation in PowerShell must have sufficient privileges in AD for this operation to complete successfully. If the current user does not have sufficient privileges, the ADUserName and ADPassword parameters may be used instead.

If the option to remove the account from AD or to disable it in AD is specified, the AD operation must succeed for the account to be removed from the Citrix AD Identity Service database. Use caution when using the Force parameter because this allows removal of accounts that are in the 'inUse'state, which may result in the machines becoming unusable.

Examples

EXAMPLE 1

Removes two accounts (account and account2) from the identity pool called "MyPool", leaving the AD accounts untouched.

```
Remove-AcctADAccount -IdentityPoolName MyPool -ADAccountName "Domain\
    account","domain\account2"


    SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
    -----------------------       -------------------    --------------
                          2                         0    {
}
```

EXAMPLE 2

Removes two accounts (account and account2) from the identity pool called "MyPool"(and from AD).

```
Remove-AcctADAccount -IdentityPoolName MyPool -RemovalOption Delete -
    ADAccountName "Domain\account","domain\account2"


    SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
    -----------------------       -------------------    --------------
                          2                         0    {
}
```


EXAMPLE 3

Removes two accounts (account and account2) from the identity pool called “MyPool”, leaving the AD accounts untouched. The accounts are removed regardless of whether they are in the ‘inUse’state or not.

```
Remove-AcctADAccount -IdentityPoolName MyPool -ADAccountName "Domain\
    account","domain\account2" -Force


SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
                      2                         0    {
  }
```

EXAMPLE 4

Shows failure of removal of one of two accounts and how to retrieve the failure reason.

```
Remove-AcctADAccount -IdentityPoolName MyPool -ADAccountName "Domain\
    account","domain\account2" -OutVariable result

SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
1                                                1    {
 account2 }


$result[0].FailedAccounts

ADAccountName                 ADAccountSid           ErrorReason
-------------                 -------------          ------------
Domain\account2               account2
    IdentityObjectLocked
```

EXAMPLE 5

Removes one account (S-1-5-21-1315084875-1285793635-2418178940-2685) from the identity pool called “MyPool”, leaving the AD accounts untouched.

```
Remove-AcctADAccount -IdentityPoolName MyPool -ADAccountSid S
    -1-5-21-1315084875-1285793635-2418178940-2685


SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
1                                                0    {
  }
```


Parameters

-IdentityPoolName

The name of the identity pool from which accounts are to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountName

The name of the AD account to be removed. AD accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g `MyComputer@MyDomain.Com`; Domain qualified e.g MyDomain\MyComputer.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountSid

The SID for the AD account to be removed.

Type:	String[]
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool from which accounts are to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ADUserName

The username for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADPassword

The matching password for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	SecureString

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemovalOption

Defines the behavior relating to the AD account.

None - Do not attempt to remove the account from AD Delete - Attempt to remove the account from AD Disable - Attempt to disable the account in AD

Type:	ADIdentityRemoveAccountOption
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

Indicates if accounts that are marked as ‘in-use’can be removed.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationSummary

The remove-AcctADAccout command returns an object with the following parameters; SuccessfulAccountsCount <int>

The number of accounts that were removed successfully.

FailedAccountsCount <int>

The number of accounts that were not removed.

FailedAccounts <Citrix.ADIdentity.Sdk.AccountError[]>

The list of accounts that failed to be removed. Each one has the following parameters:

ADAccountName <string>

ADAccountSid <String>

ErrorReason <ADIdentityStatus>

This can be one of the following

UnableToConvertDomain

IdentityNotLocatedInDomain

IdentityNotInIdentityPool

IdentityObjectInUse

IdentityObjectLocked

ADServiceDatabaseError

ADServiceDatabaseNotConfigured

ADServiceStatusInvalidDb

FailedToConnectToDomainController

FailedToDisableAccountInAD

FailedToDeleteAccountInAD

FailedToExecuteSearchInAD

FailedToAccessComputerAccountInAD

DiagnosticInformtion <Exception>

Any other error information

Notes

In the case of failure, the following errors can result.

Error Codes

IdentityPoolNotFound

The specified identity pool was not found.

IdentityPoolAlreadyLocked

The specified identity pool was locked by another operation.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Add-AcctADAccount
- Repair-AcctADAccount
- Unlock-AcctADAccount
- Update-AcctADAccount
- Get-AcctADAccount

Remove-AcctIdentityPool

March 11, 2024

Removes identity pools.

Syntax

```
Remove-AcctIdentityPool
      [-IdentityPoolName] <String>
      [-AzureADAccessToken <String>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPool
      -IdentityPoolUid <Guid>
      [-AzureADAccessToken <String>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove identity pools. The identity pool must be emptied of any AD accounts that it contains before it can be removed.

Examples

EXAMPLE 1

Removes the identity pool named “MyPool”.

```
c:\Remove-AcctIdentityPool -IdentityPoolName MyPool
```

Parameters

-IdentityPoolName

The name of the identity pool to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADAccessToken

The access token of Microsoft Graph API. Must be specified if “SecurityGroup”is managed by Citrix.

Make sure grant consent to following permissions:

- Group.ReadWrite.All
- GroupMember.ReadWrite.All

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

IdentityPoolObjectNotFound

The specified identity pool could not be located.

UnableToRemoveDueToAssociatedAccounts

The identity pool is not empty.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

AzureADTenantIdInIdentityContentMismatchAzureADAccessToken

The TenantId extracted from AccessToken must be the same as specified TenantId in IdentityContent.

FailedToDeleteAzureADSecurityGroup

An error occured in the service while deleting Azure AD security group.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Rename-AcctIdentityPool
- Set-AcctIdentityPool
- Unlock-AcctIdentityPool

Remove-AcctIdentityPoolMetadata

March 11, 2024

Removes metadata from the given identity pool.

Syntax

```
Remove-AcctIdentityPoolMetadata
      [-IdentityPoolUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolMetadata
      [-IdentityPoolUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolMetadata
      [-IdentityPoolName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolMetadata
      [-IdentityPoolName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-AcctIdentityPoolMetadata
      [-InputObject] <IdentityPool[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolMetadata
      [-InputObject] <IdentityPool[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given identity pool.

The identity pools may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Remove the metadata property “metadataName”from the identity pool “MyPool”.

```
Remove-AcctIdentityPoolMetadata -IdentityPoolName MyPool -Name
   metadataName
```

EXAMPLE 2

Removes the metadata properties “metadataName1”and “metadataName2”from the identity “MyPool”. Note that the values of the properties DO NOT have to match “val1”and “val2”.

```
Remove-AcctIdentityPoolMetadata -IdentityPoolName MyPool -Map (@{
 "metadataName1" = "val1"; "metadataname2" = "val2" }
 )
```

EXAMPLE 3

Remove all metadata from all identity pool objects.

```
Get-AcctIdentityPool | % {
  Remove-AcctIdentityPoolMetadata -Map $_.MetadataMap  }
```


Parameters

-IdentityPoolUid

The unique identifier for the identity pool from which metadata is to be removed.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool from which metadata is to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects from which the metadata is to be removed.

Type:	IdentityPool[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Set-AcctIdentityPoolMetadata

Remove-AcctIdentityPoolScope

March 11, 2024

Remove the specified identity pool(s) from the given scope(s).

Syntax

```
Remove-AcctIdentityPoolScope
      [-Scope] <String[]>
      -InputObject <IdentityPool[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolScope
      [-Scope] <String[]>
      -IdentityPoolUid <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctIdentityPoolScope
      [-Scope] <String[]>
      -IdentityPoolName <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-AcctIdentityPoolScope command is used to remove one or more identity pool objects from the given scope(s).

To remove a identity pool from a scope you need permission to change the scopes of the identity pool.

If the identity pool is not in a specified scope, that scope will be silently ignored.

The identity pools may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Removes a single identity pool from the 'Finance'scope.

```
Remove-AcctIdentityPoolScope Finance -IdentityPoolUid 6702C5D0-C073
    -4080-A0EE-EC74CB537C52
```

EXAMPLE 2

Removes a single identity pool from multiple scopes.

```
Remove-AcctIdentityPoolScope Finance,Marketing -IdentityPoolUid 6702
    C5D0-C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 3

Removes all visible identity pool objects from the 'Finance'scope.

```
Get-AcctIdentityPool | Remove-AcctIdentityPoolScope Finance
```

EXAMPLE 4

Removes identity pool objects with a name starting with an 'A'from the 'Finance'scope.

```
Remove-AcctIdentityPoolScope Finance -IdentityPoolName A*
```

Parameters

-Scope

Specifies the scopes to remove the objects from.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-InputObject

The identity pool objects to be removed.

Type:	IdentityPool[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pools to be removed.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pools to be removed.

Type:	String[]
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Add-AcctIdentityPoolScope
- Get-AcctScopedObject

Remove-AcctServiceAccount

March 11, 2024

Removes a service account.

Syntax

```
Remove-AcctServiceAccount
      -ServiceAccountUid <Guid>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove a service account.

Examples

EXAMPLE 1

Removes the service account specified by uid 17631afc-2e4c-491e-b0aa-f979a80e32c1.

```
c:\Remove-AcctServiceAccount -ServiceAccountUid 17631afc-2e4c-491e-b0aa
   -f979a80e32c1
```


Parameters

-ServiceAccountUid

The unique identifier for the service account to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

ServiceAccountObjectNotFound

The specified service account could not be located.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctServiceAccount
- Get-AcctServiceAccount
- Set-AcctServiceAccount

Remove-AcctServiceAccountScope

March 11, 2024

Remove the specified service account(s) from the given scope(s).

Syntax

```
Remove-AcctServiceAccountScope
      [-Scope] <String[]>
      -InputObject <ServiceAccount[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctServiceAccountScope
      [-Scope] <String[]>
      -ServiceAccountUid <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-AcctServiceAccountScope command is used to remove one or more service account objects from the given scope(s).

To remove a service account from a scope you need permission to change the scopes of the service account.

If the service account is not in a specified scope, that scope will be silently ignored.

The service accounts may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Removes a single service account from the 'Finance'scope.

```
Remove-AcctServiceAccountScope Finance -ServiceAccountUid 17631afc-2e4c
    -491e-b0aa-f979a80e32c1
```

EXAMPLE 2

Removes a single service account from multiple scopes.

```
Remove-AcctServiceAccountScope Finance,Marketing -ServiceAccountUid
    17631afc-2e4c-491e-b0aa-f979a80e32c1
```

EXAMPLE 3

Removes all visible service account objects from the 'Finance'scope.

```
Get-AcctServiceAccount | Remove-AcctServiceAccountScope Finance
```

Parameters

-Scope

Specifies the scopes to remove the objects from.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-InputObject

The service account objects to be removed.

Type:	ServiceAccount[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ServiceAccountUid

The unique identifier for the service accounts to be removed.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Add-AcctServiceAccountScope
- Get-AcctScopedObject

Remove-AcctServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-AcctServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AcctServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.


```
Get-AcctService | % {
  Remove-AcctServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Set-AcctServiceMetadata

Rename-AcctIdentityPool

March 11, 2024

Renames an identity pool.

Syntax

```
Rename-AcctIdentityPool
      [-IdentityPoolName] <String>
      [-NewIdentityPoolName] <String>
      [-PassThru]
```



```
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Rename-AcctIdentityPool
       -IdentityPoolUid <Guid>
       [-NewIdentityPoolName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to change the name of an existing identity pool.

Examples

EXAMPLE 1

Renames an existing identity pool called “oldName”to be called “newName”.

```
Rename-AcctIdentityPool -IdentityPoolName oldName -NewIdentityPoolName
    newName
```

Parameters

-IdentityPoolName

The name of the identity pool to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool to be renamed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewIdentityPoolName

The new name for the identity pool. This must be a name which is not used by an existing identity pool, and it must not contain any of the following characters \/;:#.*?=<>|[]()’””

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PassThru

Defines whether or not the command returns a result showing the new state of the updated provisioning scheme.

Type:	SwitchParameter
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	SwitchParameter
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

AvailableAccounts <int>

The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse', 'Tainted', or 'Error').

DeviceManagementType <string>

The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

Domain <string>

The Active Directory domain (in FQDN format) that accounts in the pool belong to.

ErrorAccounts <int>

The number of existing AD accounts in the 'Error'state.

IdentityContent <string>

JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

IdentityPoolName <string>

The name of the identity pool.

IdentityPoolUid <GUID>

The unique identifier for the identity pool.

IdentityType <string>

The identity type.

InUseAccounts <int>

The number of existing AD accounts in the 'InUse' state.

Lock <bool>

Indicates if the identity pool is locked.

MetadataMap <IDictionary[string, string];>

The metadata associated with this identity pool arranged in key value pairs.

NamingScheme <string>

The naming scheme for the identity pool.

NamingSchemeType <string>

The naming scheme type for the identity pool. This can be one of the following:

Numeric - naming scheme uses numeric indexes

Alphabetic - naming scheme uses alphabetic indexes

OU <string>

The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

ResourceLocationId <GUID>

The UID that corresponds to the resource location (DaaS only).

StartCount <int>

The next index to be used when creating an account in the identity pool.

TaintedAccounts <int>

The number of existing AD accounts in the 'Tainted' state.

WorkgroupMachine <bool>

If this is true, the identity pool can have an IdentityType of 'AzureAD' or 'Workgroup'.

If this is false, the identity pool can have an IdentityType of 'ActiveDirectory' or 'HybridAzureAD'.

ZoneUid <GUID>

The UID that corresponds to the Zone in which AD accounts are created.

Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

The administration scopes associated with this identity pool.

TenantId <GUID>

Identity of the Citrix tenant associated with this identity pool.

Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure the following errors can result.

Error Codes

IdentityPoolObjectNotFound

The specified identity pool could not be located.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctIdentityPool
- Set-AcctIdentityPool
- Remove-AcctIdentityPool
- Test-AcctIdentityPoolNameAvailable

Repair-AcctADAccount

March 11, 2024

Resets the Active Directory (AD) password for the given accounts.

Syntax

```
Repair-AcctADAccount
      [-Password <String>]
      [-SecurePassword <SecureString>]
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-Force]
      [-ResetOnly]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Repair-AcctADAccount
      -ADAccountName <String[]>
      [-Password <String>]
      [-SecurePassword <SecureString>]
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
      [-Force]
      [-ResetOnly]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Repair-AcctADAccount
      -ADAccountSid <String[]>
      [-Password <String>]
      [-SecurePassword <SecureString>]
      [-ADUserName <String>]
      [-ADPassword <SecureString>]
```



```
        [-Force]
        [-ResetOnly]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This provides the ability to repair accounts in the 'Tainted'state by synchronizing the account password stored in Active Directory (AD) with the password stored in the AD Identity Service. If successful, this results in the account state being reset to 'Available'so it can be consumed by other Machine Creation Services.

If the current account password is not supplied using the Password or SecurePassword Parameters, this requires the user who initiated the runspace to have the required permissions in AD to reset the AD account password. If the current user does not have sufficient privileges, the ADUserName and ADPassword parameters may be used instead.

If the current account password is supplied then this command will use the password change operation which does not require any elevated permissions in AD.

Examples

EXAMPLE 1

Repairs the tainted accounts account and account2. After the repair operation, there are no more tainted accounts.

```
Get-AcctADAccount -State Tainted | Select-Object ADAccountName

ADAccountName
-------------
Domain\account
Domain\account2

Repair-AcctADAccount -ADAccountName "Domain\account","Domain\account2"

SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
                      2                         0    {
  }


Get-AcctADAccount -State Tainted
```


Parameters

-ADAccountName

The names of the AD accounts that are to be repaired. AD accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g `MyComputer@MyDomain.Com`; Domain qualified e.g MyDomain\MyComputer.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountSid

The SID(s) of the accounts that are to be repaired.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Password

The current password for the computer account.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SecurePassword

The current password for the account (provided in a Secure String class).

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADUserName

The username for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADPassword

The matching password for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	SecureString

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

Indicates whether accounts that are marked as ‘in-use’can be repaired or not.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResetOnly

Indicates whether only resetting machine password and trust keys without updating the states to ‘Available’if the accounts that are marked as ‘InUse’. This parameter will be sliently ignored by the accounts which states are not ‘InUse’.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationSummary

The Repair-AcctADAccout command returns an object with the following parameters: SuccessfulAccountsCount <int>

The number of accounts that were repaired successfully.

FailedAccountsCount <int>

The number of accounts that were not repaired.

FailedAccounts <Citrix.ADIdentity.Sdk.AccountError[]>

The list of accounts that failed to be repaired. Each one has the following parameters:

ADAccountName <string>

ADAccountSid <String>

ErrorReason <ADIdentityStatus>

This can be one of the following

UnableToConvertDomain

IdentityNotLocatedInDomain

IdentityNotFound

IdentityObjectInUse

IdentityObjectLocked

ADServiceDatabaseError

ADServiceDatabaseNotConfigured

ADServiceStatusInvalidDb

FailedToConnectToDomainController

FailedToExecuteSearchInAD

FailedToAccessComputerAccountInAD

FailedToSetPasswordInAD

FailedToChangePasswordInAD

DiagnosticInformtion <Exception>

Any other error information

Notes

In the case of failure, the following errors can result.

Error Codes

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Add-AcctADAccount
- Remove-AcctADAccount
- Unlock-AcctADAccount
- Update-AcctADAccount
- Get-AcctADAccount

Repair-AcctIdentity

March 11, 2024

Repair the given identity accounts in identity pool.

Syntax

```
Repair-AcctIdentity
      -Target <String>
      [-PrivilegedUserName <String>]
      [-PrivilegedUserPassword <SecureString>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Repair-AcctIdentity
      -IdentityAccountName <String[]>
      -Target <String>
      [-PrivilegedUserName <String>]
      [-PrivilegedUserPassword <SecureString>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Repair-AcctIdentity
      -IdentityAccountId <String[]>
      -Target <String>
      [-PrivilegedUserName <String>]
      [-PrivilegedUserPassword <SecureString>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This provides the ability to repair accounts in the identity pool that behave abnormal without changing the current states of the accounts. Unlike "Repair-AcctADAccount", this command will not reset 'Tainted'accounts and make them to be 'Available'in the identity pool.

Examples

EXAMPLE 1

Repairs userCertificate attributes for the Hybrid Azure AD joined identity accounts in the identity pool "MyIdentityPool".

```
Get-AcctADAccount -IdentityPoolName "MyIdentityPool" | Repair-
    AcctIdentity -Target UserCertificate

SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
                      2                         0    {
```



```
6    }
```

Parameters

-IdentityAccountName

The names of the identity accounts that are to be repaired. AD based identity accounts are accepted in the following formats: Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com; UPN format e.g `MyComputer@MyDomain.Com`; Domain qualified e.g MyDomain\MyComputer.

Type:	String[]
Aliases:	ADAccountName
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityAccountId

The ID(s) of the identity accounts that are to be repaired. For AD based identity account, it should be SID

Type:	String[]
Aliases:	ADAccountSid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Target

The target to be repaired. It can be either ‘IdentityInfo’or ‘UserCertificate’. Specifies ‘IdentityInfo ‘to repair account password and trust key pair. Specifies ‘UserCertificate’to repair userCertificate for

Hybrid Azure AD joined identities.

Type:	String
Accepted values:	IdentityInfo, UserCertificate
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PrivilegedUserName

The username for an user account in the identity provider (e.g. AD or Azure AD) with Write Permissions. This parameter must be used if the current user does not have the necessary privileges to modify accounts in identity provider.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrivilegedUserPassword

The matching password for an user account in the identity provider (e.g. AD or Azure AD) with Write Permissions. This parameter must be used if the current user does not have the necessary privileges to modify accounts in identity provider.

Type:	SecureString
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationSummary

The Repair-AcctIdentity command returns an object with the following parameters: SuccessfulAccountsCount <int>

The number of accounts that were repaired successfully.

FailedAccountsCount <int>

The number of accounts that were not repaired.

FailedAccounts <Citrix.ADIdentity.Sdk.AccountError[]>

The list of accounts that failed to be repaired. Each one has the following parameters:

ADAccountName <string>

ADAccountSid <String>

ErrorReason <ADIdentityStatus>

This can be one of the following

UnableToConvertDomain

IdentityNotLocatedInDomain

IdentityNotFound

IdentityObjectInUse

IdentityObjectLocked

ADServiceDatabaseError

ADServiceDatabaseNotConfigured

ADServiceStatusInvalidDb

FailedToConnectToDomainController

FailedToExecuteSearchInAD

FailedToAccessComputerAccountInAD

FailedToSetPasswordInAD

FailedToChangePasswordInAD

DiagnosticInformtion <Exception>

Any other error information

Notes

In the case of failure, the following errors can result.

Error Codes

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Repair-AcctADAccount

Reset-AcctEnabledFeatureList

March 11, 2024

Refreshes the ADIdentity service's list of enabled features.

Syntax

```
Reset-AcctEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix ADIdentity Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected ADIdentity service instance's list of enabled features.

```
Reset-AcctEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-AcctServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the ADIdentity Service.

Syntax

```
Reset-AcctServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload ADIdentity Service access permissions and configuration service locations. The Reset-AcctServiceGroupMembership command must be run on at least one instance of the service type (Acct) after installation and registration with the configuration service. Without this operation, the ADIdentity services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AcctServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    AcctServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-AcctServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.ADIdentity.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AcctServiceGroupMembership command.

Outputs

Citrix.ADIdentity.Sdk.ServiceInstance

Reset-AcctServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the ADIdentity Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctServiceInstance
- Get-AcctServiceStatus

Set-AcctADAccountUserCert

March 11, 2024

Set userCertificate for the given Active Directory (AD) accounts.

Syntax

```
Set-AcctADAccountUserCert
   [-IdentityPoolName] <String>
   -ADAccountName <String[]>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Force]
   [-BatchSize <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctADAccountUserCert
   [-IdentityPoolName] <String>
   -ADAccountSid <String[]>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Force]
   [-BatchSize <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctADAccountUserCert
   [-IdentityPoolName] <String>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-AllAccounts]
   [-Force]
   [-BatchSize <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctADAccountUserCert
   -IdentityPoolUid <Guid>
   -ADAccountName <String[]>
   [-ADUserName <String>]
   [-ADPassword <SecureString>]
   [-Force]
   [-BatchSize <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctADAccountUserCert
   -IdentityPoolUid <Guid>
   -ADAccountSid <String[]>
```



```
    [-ADUserName <String>]
    [-ADPassword <SecureString>]
    [-Force]
    [-BatchSize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AcctADAccountUserCert
   -IdentityPoolUid <Guid>
    [-ADUserName <String>]
    [-ADPassword <SecureString>]
    [-AllAccounts]
    [-Force]
    [-BatchSize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to set the userCertificate attribute for given accounts in Active Directory (AD).

The user performing the operation in PowerShell must have sufficient privileges in AD to set the userCertificate attribute. If the current user does not have sufficient privileges, the ADUserName and ADPassword parameters may be used instead.

Examples

EXAMPLE 1

Set userCertificate for two accounts (account and account2) in the identity pool called “MyPool”.

```
Set-AcctADAccountUserCert -IdentityPoolName MyPool -ADAccountName "
   Domain\account","Domain\account2"


SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
-----------------------       -------------------    --------------
2                          0    {
  }
```

EXAMPLE 2

Set userCertificate for two accounts (account and account2) in the identity pool called “MyPool”. The accounts are set regardless of whether they are in the ‘inUse’state or not.


```
Set-AcctADAccountUserCert -IdentityPoolName MyPool -ADAccountName "
    Domain\account","Domain\account2" -Force


  SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
  -----------------------       -------------------    --------------
  2                          0    {
  }
```

EXAMPLE 3

Shows failure of setting of one of two accounts and how to retrieve the failure reason.

```
Set-AcctADAccountUserCert -IdentityPoolName MyPool -ADAccountName "
    Domain\account","Domain\account2" -OutVariable result

  SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
  -----------------------       -------------------    --------------
  1                                                 1    {
 account2 }


  C:\PS>$result[0].FailedAccounts

ADAccountName                 ADAccountSid           ErrorReason
-------------                 -------------          ------------
Domain\account2               account2
    IdentityObjectLocked
```

EXAMPLE 4

Set userCertificate for one account (S-1-5-21-1315084875-1285793635-2418178940-2685) in the identity pool called “MyPool”.

```
Set-AcctADAccountUserCert -IdentityPoolName MyPool -ADAccountSid S
    -1-5-21-1315084875-1285793635-2418178940-2685


  SuccessfulAccountsCount       FailedAccountsCount    FailedAccounts
  -----------------------       -------------------    --------------
  1                                                 0    {
  }
```


Parameters

-IdentityPoolName

The name of the identity pool containing the accounts to have userCertificate set.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountName

The AD account name to be set with userCertificate.

Active Directory accounts are accepted in the following formats:

Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com;

UPN format e.g `MyComputer@MyDomain.Com`;

Domain qualified e.g MyDomain\MyComputer.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountSid

The SID for the AD account to be set with userCertificate.

Type:	String[]

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AllAccounts

Indicates if all accounts should be set with userCertificate.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier of the identity pool containing the accounts to have userCertificate set.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ADUserName

The username for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ADPassword

The matching password for an AD user account with Write Permissions. This parameter must be used if the current user does not have the necessary privileges.

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

Indicates if accounts that are marked as 'in-use'can be set with userCertificate.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BatchSize

Specifies the number of accounts to set user certificate in one batch to AD Identity Service.

Type:	Int32
Position:	Named
Default value:	50
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.AccountOperationSummary

The Set-AcctADAccountUserCert command returns an object with the following parameters: SuccessfulAccountsCount <int>

The number of accounts that were successfully set with userCertificate.

FailedAccountsCount <int>

The number of accounts that were failed to set with userCertificate.

FailedAccounts <Citrix.ADIdentity.Sdk.AccountError[]>

The list of accounts that failed to be set with userCertificate. Each one has the following parameters:

ADAccountName <string>

ADAccountSid <String>

ErrorReason <ADIdentityStatus>

This can be one of the following:

IdentityObjectNotFound

IdentityObjectLocked

IdentityObjectInUse

IdentityPoolObjectNotFound

ADServiceDatabaseError

ADServiceDatabaseNotConfigured

ADServiceStatusInvalidDb

FailedToConnectToDomainController

FailedToGenerateDeviceKeyPair

FailedToAccessComputerAccountInAD

UnauthorizedToWriteUserCertificateInAD

DiagnosticInformation <Exception>

Any other error information

Notes

In the case of failure, the following errors can result.

Error Codes

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Add-AcctADAccount
- Repair-AcctADAccount

- Unlock-AcctADAccount
- Update-AcctADAccount
- Get-AcctADAccount

Set-AcctDBConnection

March 11, 2024

Configures a database connection for the ADIdentity Service.

Syntax

```
Set-AcctDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix ADIdentity Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-AcctDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-AcctDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the ADIdentity Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-AcctDBConnection cmdlet returns an object describing the status of the ADIdentity Service together with extra diagnostics information. Possible values are:

- OK:

The ADIdentity Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the ADIdentity Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the ADIdentity Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the ADIdentity Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the ADIdentity Service currently in use is newer than, and incompatible with, the version of the ADIdentity Service schema on the database. Upgrade the ADIdentity Service to a more recent version.

- DBOlderVersionThanService:

The version of the ADIdentity Service schema on the database is newer than, and incompatible with, the version of the ADIdentity Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the ADIdentity Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the ADIdentity Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the ADIdentity Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctServiceStatus
- Get-AcctDBConnection
- Test-AcctDBConnection

Set-AcctDBCredentials

March 11, 2024

Configures the database server SQL credentials for the ADIdentity Service.

Syntax

```
Set-AcctDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Specifies SQL credentials to be used by the currently selected Citrix ADIdentity Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional ADIdentity Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-AcctDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-AcctDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-AcctDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.


```
Set-AcctDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AcctDBSchema
- Set-AcctDBConnection
- Get-Credential

Set-AcctIdentityPool

March 11, 2024

Update parameters of an identity pool.

Syntax

```
Set-AcctIdentityPool
   [-IdentityPoolName] <String>
   [-NamingScheme <String>]
   [-NamingSchemeType <ADIdentityNamingScheme>]
   [-OU <String>]
   [-Domain <String>]
   [-AllowUnicode]
   [-PassThru]
   [-StartCount <Int32>]
   [-ZoneUid <Guid>]
   [-AzureADSecurityGroupName <String>]
   [-AzureADAccessToken <String>]
   [-AzureADTenantId <Guid>]
   [-ServiceAccountUid <Guid[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctIdentityPool
   -IdentityPoolUid <Guid>
   [-NamingScheme <String>]
   [-NamingSchemeType <ADIdentityNamingScheme>]
   [-OU <String>]
   [-Domain <String>]
```



```
    [-AllowUnicode]
    [-PassThru]
    [-StartCount <Int32>]
    [-ZoneUid <Guid>]
    [-AzureADSecurityGroupName <String>]
    [-AzureADAccessToken <String>]
    [-AzureADTenantId <Guid>]
    [-ServiceAccountUid <Guid[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to modify the parameters of an identity pool.

Examples

EXAMPLE 1

Changes the start count and naming scheme of the identity pool named ‘poolName’so that the next account generated will be AC0100 (assuming that account does not already exist).

```
Set-AcctIdentityPool -IdentityPoolName poolName -StartCount 100 -
    NamingScheme AC####
```

Parameters

-IdentityPoolName

The name of the identity pool that is to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool that is to be modified.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NamingScheme

The new naming scheme that is to be used for the identity pool.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NamingSchemeType

The new naming scheme type that is to be used for the identity pool. This can be Numeric or Alphabetic.

Type:	ADIdentityNamingScheme
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OU

The new OU to be used for the identity pool. All accounts created after this is set are created in this AD container. This will not move existing accounts. The OU must be a valid AD container of the domain specified for the pool.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Domain

The new Active Directory domain that is to be used for the identity pool. All new accounts will be created in this domain, but will not affect existing accounts. The domain can be specified in either long or short form (i.e. domain or domain.com).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowUnicode

Updates the definition of the allowed characters in a naming scheme.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Defines whether the command returns the new state of the identity pool or not.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartCount

Specifies the next number to be used when creating new AD accounts in the identity pool.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

The UID that corresponds to the Zone in which these AD accounts will be created. This is only intended to be used for Citrix Cloud Delivery Controllers.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADSecurityGroupName

The name of Azure AD security group. This is only intended to be used when IdentityType is “AzureAD” or “HybridAzureAD”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADAccessToken

The access token of Microsoft Graph API. Must be specified if “IdentityType”is “HybridAzureAD”or “AzureAD”.

Make sure grant consent to following permissions:

- Group.ReadWrite.All
- GroupMember.ReadWrite.All

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADTenantId

The tenantId of AzureAD. Must be specified if “AzureADSecurityGroupName”is specified to create AzureAD security group, and IdentityType is “AzureAD”or “HybridAzureAD”.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceAccountUid

Specify the UID of the service account that associates with the identity pool.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.IdentityPool

This object provides details of the identity pool and contains the following information:

AvailableAccounts <int>

The number of existing accounts (AcctADAccount objects) in the 'Available'state (not in 'InUse', 'Tainted', or 'Error').

DeviceManagementType <string>

The device management type. Can be Intune, IntuneWithCitrixTags, or None by default.

Domain <string>

The Active Directory domain (in FQDN format) that accounts in the pool belong to.

ErrorAccounts <int>

The number of existing AD accounts in the 'Error'state.

IdentityContent <string>

JSON formatted metadata containing Azure AD tenant and Azure AD security group information associated with this identity pool.

IdentityPoolName <string>

The name of the identity pool.

IdentityPoolUid <GUID>

The unique identifier for the identity pool.

IdentityType <string>

The identity type.

InUseAccounts <int>

The number of existing AD accounts in the 'InUse' state.

Lock <bool>

Indicates if the identity pool is locked.

MetadataMap <IDictionary[string, string];>

The metadata associated with this identity pool arranged in key value pairs.

NamingScheme <string>

The naming scheme for the identity pool.

NamingSchemeType <string>

The naming scheme type for the identity pool. This can be one of the following:

Numeric - naming scheme uses numeric indexes

Alphabetic - naming scheme uses alphabetic indexes

OU <string>

The Active Directory distinguished name for the OU in which accounts for this identity pool will be created.

ResourceLocationId <GUID>

The UID that corresponds to the resource location (DaaS only).

StartCount <int>

The next index to be used when creating an account in the identity pool.

TaintedAccounts <int>

The number of existing AD accounts in the 'Tainted' state.

WorkgroupMachine <bool>

If this is true, the identity pool can have an IdentityType of 'AzureAD'or 'Workgroup'.

If this is false, the identity pool can have an IdentityType of 'ActiveDirectory'or 'HybridAzureAD'.

ZoneUid <GUID>

The UID that corresponds to the Zone in which AD accounts are created.

Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]>

The administration scopes associated with this identity pool.

TenantId <GUID>

Identity of the Citrix tenant associated with this identity pool.

Not applicable (always blank) in non-multitenant sites.

AzureADSecurityGroupName <string>

Associate an AzureAD security group to the identity pool. When the identity pool Associated an AzureAd device security

group, the option will update the name of the associated Azure AD security group.

Notes

In the case of failure, the following errors can result.

Error Codes

InvalidIdentityPoolParameterCombination

Caused by either of the following validation errors:

- If an OU is specified then a domain must also be specified.
- NamingScheme, NamingSchemeType and Domain must all be present if any of them are specified.

OUParameterRequiresDomain

If an OU is specified then a domain must also be specified.

OUNotReachable

Domain is not reachable. Check if the domain exists in any zones and if the domain contains the specified OU.

NamingSchemeAndNamingSchemeTypeMustBeUsedTogether

NamingScheme and NamingSchemeType must be used together.

NamingSchemeIllegalComputerName

The naming scheme supplied is not valid.

UnableToConvertDomainName

Unable to convert domain name to DNS format.

NamingSchemeNotEnoughCharacters

Naming scheme does not have enough characters specified.

NamingSchemeTooManyCharacters

Naming scheme has too many characters specified.

NamingSchemeIllegalCharacter

Naming scheme contains illegal characters.

NamingSchemeMayNotStartWithPeriod

Naming scheme starts with a period (.) character.

NamingSchemeMayNotBeAllNumbers

Naming scheme contains only numbers.

NamingSchemeMissingNumericSpecifications

Naming scheme does not contain any variable specification (i.e. no '#'characters are specified).

NamingSchemeHasMoreThanOneSetOfHashes

Naming scheme has more than one variable region (i.e. there are '#'characters separated by other characters).

IdentityPoolDuplicateObjectExists

An identity pool with the same name exists already.

IdentityPoolObjectNotFound

The identity pool to be modified could not be located.

IdentityPoolOUInvalid

Identity pool OU invalid as it does not exist.

IdentityPoolOUOfWrongDomain

Identity pool OU invalid as it refers to a different domain to the domain specified for the pool.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

AzureADAccessTokenMustBeSpecified

Must specify “AzureADAccessToken”parameter to operate AzureAD security group.

AzureADSecurityGroupNotSupportCurrentIdentityType

Current IdentityType not support AzureAD security group.

AzureADSecurityGroupMustBeSpecified

Must specify AzureAD security group.

AzureADSecurityGroupMembershipRulesMoreThanFiveExpressions

AzureAD security group membership rules more than five expressions.

AzureADSecurityGroupMembershipRuleExceedsMaximumLength

AzureAD security group membership rule exceeds maximum length.

AzureADSecurityGroupMembershipNeedsMatchRules

AzureAD security group membership needs match rules.

AzureADSecurityGroupMembershipRulesPrepareFailed

AzureAD security group membership rules prepare failed.

AzureADTenantIdMismatchAzureADAccessToken

The given AzureADTenantId mismatchs TenantId contained in AzureAD AccessToken.

CurrentIdentityTypeRequiresWorkgroupMachine

Current IdentityType requires WorkgroupMachine.

DomainNotRequiredForAzureADAndWorkgroup

Domain not required for AzureAD and Workgroup.

FailedToUpdateAzureADSecurityGroup

Failed to update AzureAD security group.

NamingSchemeAndDomainMustBeUsedTogetherForActiveDirectoryOrHybridAzureAD

NamingScheme and Domain must be used together for ActiveDirectory or HybridAzureAD.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Get-AcctIdentityPool
- Remove-AcctIdentityPool

Set-AcctIdentityPoolMetadata

March 11, 2024

Adds or updates metadata on the given identity pool.

Syntax

```
Set-AcctIdentityPoolMetadata
   [-IdentityPoolUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctIdentityPoolMetadata
   [-IdentityPoolUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-AcctIdentityPoolMetadata
   [-IdentityPoolName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctIdentityPoolMetadata
   [-IdentityPoolName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctIdentityPoolMetadata
   [-InputObject] <IdentityPool[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctIdentityPoolMetadata
   [-InputObject] <IdentityPool[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given identity pool objects.

The identity pools may be specified using the unique identifier, name, or as a powershell object.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the identity pool with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-AcctIdentityPoolMetadata -IdentityPoolUid 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value
```


Parameters

-IdentityPoolUid

The unique identifier for the identity pool to which metadata is to be added.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool to which metadata is to be added.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Identity pool objects to which the metadata is to be added.

Type:	IdentityPool[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the identity pool specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1"= "val1"; "name2"= "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

Type:	PSObject

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AcctIdentityPoolMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctIdentityPoolMetadata

Set-AcctServiceAccount

March 11, 2024

Update parameters of a service account.

Syntax

```
Set-AcctServiceAccount
   -ServiceAccountUid <Guid>
   [-AccountId <String>]
   [-AccountSecret <SecureString>]
   [-SecretExpiryTime <DateTime>]
   [-Capabilities <String[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Provides the ability to modify the parameters of a service account.

Examples

EXAMPLE 1

Changes the AccountId for service account specified by uid 17631afc-2e4c-491e-b0aa-f979a80e32c1.

```
Set-AcctServiceAccount -ServiceAccountUid 17631afc-2e4c-491e-b0aa-
    f979a80e32c1 -AccountId deb0811e-4839-4cce-87d3-8f36b31c2934

                    ServiceAccountUid            : 17631afc-2e4c-491e-
                        b0aa-f979a80e32c1
                    IdentityProviderIdentifier   : f439f4c0-fcd8-4fe6
                        -95b8-71e7e49dc8c6
                    IdentityProviderType         : AzureAD
                    SecretExpiryTime             : 9/8/2024 8:00:00 PM
                    AccountId                    : deb0811e-4839-4cce
                        -87d3-8f36b31c2934
                    Capabilities                 : {
 AzureArcResourceManagement }

                    FailureReason                :
                    IsHealthy                    : True
                    Scopes                       : {
  }

                    TenantId                     :
```

EXAMPLE 2

Changes the AccountSecret for service account specified by uid ad24284e-ba3d-4504-80db-9ac6640de533.

```
Set-AcctServiceAccount -ServiceAccountUid ad24284e-ba3d-4504-80db-9
    ac6640de533 -AccountSecret $newPassword

ServiceAccountUid            : ad24284e-ba3d-4504-80db-9ac6640de533
IdentityProviderIdentifier   : test.local
IdentityProviderType         : ActiveDirectory
SecretExpiryTime             : 9/8/2024 8:00:00 PM
AccountId                    : test\svcacct_1
Capabilities                 :
FailureReason                :
IsHealthy                    : True
Scopes                       : {
```



```
12    }
13
14  TenantId                    :
```

Parameters

-ServiceAccountUid

The unique identifier for the service account that is to be modified.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AccountId

The identifier for the service account.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AccountSecret

The secret for the service account. E.g. Azure application (client) secret if 'IdentityProviderType'is AzureAD. The secret will be encrypted and stored in database.

Type:	SecureString

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecretExpiryTime

The secret expiration time for the service account.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Capabilities

The capabilities for the service account.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ADIdentity.Sdk.ServiceAccount

This object provides details of the service account and contains the following information:

ServiceAccountUid <GUID> The unique identifier of the service account. SecretExpiryTime <Date-time> The expiration time for the secret of the service account. AccountId <string> The identifier for the service account. E.g. Azure application ID if the service account is with Azure AD as identity provider. IdentityProviderIdentifier <string> The identifier of the identity provider that the service

account belongs to. E.g. Azure AD tenant ID. IdentityProviderType <string> The type of the identity provider of the service account. Can be AzureAD or ActiveDirectory. IsHealthy <bool> Indicates if the service account is healthy. Capabilities <string[]> Capabilities of the service account. Can be AzureArcResourceManagement. FailureReason <string> The reason why the service account becomes unhealthy. Scopes <Citrix.ADIdentity.Sdk.ScopeReference[]> The administration scopes associated with this identity pool. TenantId <GUID> Identity of the Citrix tenant associated with this identity pool. Not applicable (always blank) in non-multitenant sites.

Notes

In the case of failure, the following errors can result.

Error Codes

ServiceAccountObjectNotFound

The specified service account could not be located.

InvalidServiceAccountCapabilities

One or more specified service account capabilities are not supported.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctServiceAccount
- Remove-AcctServiceAccount
- Get-AcctServiceAccount

Set-AcctServiceAccountCapabilityEffectiveScope

March 11, 2024

Sets effective scope for a capability of the service account.

Syntax

```
Set-AcctServiceAccountCapabilityEffectiveScope
   -ServiceAccountUid <Guid>
   [-Capability <String>]
   [-EffectiveScope <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to set effective scope for a capability of the service account.

Examples

EXAMPLE 1

Set the effective scope for the capability AzureArcResourceManagement of service account specified by uid 17631afc-2e4c-491e-b0aa-f979a80e32c1 to a specific subscription 04b4abae-a80a-47c2-8156-daa28145e4fa.


```
c:\Set-AcctServiceAccountCapabilityEffectiveScope -ServiceAccountUid
    17631afc-2e4c-491e-b0aa-f979a80e32c1 -Capability
    AzureArcResourceManagement -EffectiveScope 04b4abae-a80a-47c2-8156-
    daa28145e4fa
```

Parameters

-ServiceAccountUid

The unique identifier for the service account.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Capability

The capability for the service account to be set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EffectiveScope

The effective scope for the capability of the service account to be set. Effective scope is the scope that the capability applies to.

For Azure AD, it can be the subscription id within the Azure AD tenant.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

ServiceAccountObjectNotFound

The specified service account could not be located.

ServiceAccountCapabilityNotFound

The specified service account capability is not found.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctServiceAccount
- Get-AcctServiceAccount
- Set-AcctServiceAccount

Set-AcctServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-AcctServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AcctServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AcctServiceMetadata
    [-InputObject] <Service[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-AcctServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AcctServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Remove-AcctServiceMetadata

Test-AcctDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix ADIdentity Service.

Syntax

```
Test-AcctDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
5       [<CommonParameters>]
```

```
1 Test-AcctDBConnection
2     [-DBConnection] <String>
3     [-Credentials] <PSCredential>
4     [-LoggingId <Guid>]
5     [<CitrixCommonParameters>]
6     [<CommonParameters>]
```

```
1 Test-AcctDBConnection
2     [-DBConnection] <String>
3     [-Login] <String>
4     [-Password] <SecureString>
5     [-LoggingId <Guid>]
6     [<CitrixCommonParameters>]
7     [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix ADIdentity Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ADIdentity SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1 Test-AcctDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
     Trusted_Connection=True"
```


Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix ADIdentity Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-AcctDBConnection cmdlet returns an object describing the status of the selected ADIdentity Service instance that would result if the connection string were used with the Set-AcctDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-AcctDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the ADIdentity Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the ADIdentity Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The ADIdentity Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The ADIdentity Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the ADIdentity Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the ADIdentity Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctServiceStatus
- Get-AcctDBConnection
- Set-AcctDBConnection

Test-AcctIdentityPoolNameAvailable

March 11, 2024

Checks to ensure that the proposed name for an identity pool is unused.

Syntax

```
Test-AcctIdentityPoolNameAvailable
    [-IdentityPoolName] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Checks to ensure that the proposed name for an identity pool is unused. This check is done without regard for scoping of existing identity pools, so the names of identity pools under inaccessible administrative scopes are also checked.

Examples

EXAMPLE 1

Tests whether the identity pool names "MyPool", "NewPool", and "UnusedPool"are available. The output indicates that only "UnusedPool"is available.

```
Test-AcctIdentityPoolNameAvailable -IdentityPoolName MyPool, NewPool,
   UnusedPool

Name       Available
----       ---------
MyPool         False
NewPool        False
UnusedPool      True
```

Parameters

-IdentityPoolName

The name or names of the identity pool(s) to be tested.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

object[]

An array of PSObjects that pair the name and availability of the name

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Rename-AcctIdentityPool

Unlock-AcctADAccount

March 11, 2024

Unlocks Active Directory (AD) accounts within the AD Identity Service.

Syntax

```
Unlock-AcctADAccount
       -ADAccountName <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Unlock-AcctADAccount
       -ADAccountSid <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to unlock Active Directory (AD) accounts within the AD Identity Service. An AD account is marked as locked in the AD Identity Service while the Machine Creation Services (MCS) are processing tasks relating to the account. If these tasks are forcibly stopped, an account can remain locked despite no longer being processed. This command resolves this issue, but use it with caution because unlocking an account that MCS expects to be locked can result in an MCS operation being cancelled. Use this command only when MCS has locked an account for use in a provisioning operation and the operation has failed without unlocking the account.

Note: The lock state in Active Directory is unrelated to the lock state in the AD Identity service. This command does NOT make any changes to the account information stored in Active Directory, only the AD Identity Service database.

Examples

EXAMPLE 1

Unlocks the AD account called "Domain\account".

```
Unlock-AcctADAccount -ADAccountName Domain\account
```

EXAMPLE 2

Unlocks all the locked AD accounts.

```
Get-AcctADAccount -Filter {
 Lock -eq $true }
  | Unlock-AcctADAccount
```

Parameters

-ADAccountName

The name of the AD account to be unlocked.

AD account name is accepted in the following formats:

Fully qualified DN e.g. CN=MyComputer,OU=Computers,DC=MyDomain,DC=Com;

UPN format e.g `MyComputer@MyDomain.Com`;

Domain qualified e.g MyDomain\MyComputer.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountSid

The SID of the AD account to be unlocked.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.ADIdentity.Sdk.IdentityInPool

You can pipe an object containing a parameter called ‘ADAccountSID’to unlock-AcctADAccount.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

IdentityNotLocatedInDomain

The specified AD account could not be located in Active Directory.

IdentityObjectNotFound

The identity could not be found.

IdentityAlreadyUnlocked

The identity is not locked.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- Get-AcctADAccount
- New-AcctADAccount
- Add-AcctADAccount
- Repair-AcctADAccount
- Remove-AcctADAccount
- Update-AcctADAccount
- Unlock-AcctADAccount

Unlock-AcctIdentityPool

March 11, 2024

Unlocks identity pools.

Syntax

```
Unlock-AcctIdentityPool
       [-IdentityPoolName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Unlock-AcctIdentityPool
       -IdentityPoolUid <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Provides the ability to unlock the specified identity pool. Identity pools are locked automatically when being updated (e.g. when new accounts are being created). The pool must never be left in a locked state; this command allows recovery from an error should one ever occur. Use this command with caution, as unlocking an identity pool which is supposed to be locked may result in unexpected behavior.

Examples

EXAMPLE 1

Unlocks the identity pool called "MyPool", setting the Lock field to false.

```
Unlock-AcctIdentityPool -IdentityPool MyPool
```

EXAMPLE 2

Unlocks all the locked identity pools.

```
Get-AcctIdentityPool -Filter {
 Lock -eq $true }
  | Unlock-AcctIdentityPool
```

Parameters

-IdentityPoolName

The name of the identity pool to be unlocked.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool to be unlocked.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.ADIdentity.Sdk.IdentityPool

You can pipe an object containing a parameter called 'IdentityPoolName' to unlock-AcctIdentityPool.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

IdentityPoolObjectNotFound

The specified identity pool could not be located.

IdentityPoolAlreadyUnlocked

The specified identity pool is not locked.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctIdentityPool
- Remove-AcctIdentityPool
- Set-AcctIdentityPool

Update-AcctADAccount

March 11, 2024

Updates the state of Active Directory (AD) accounts.

Syntax

```
Update-AcctADAccount
      [-IdentityPoolName] <String>
      [-AllAccounts]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Update-AcctADAccount
      -IdentityPoolUid <Guid>
      [-AllAccounts]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to synchronize the state of the Active Directory (AD) accounts stored in the AD Identity Service with the AD accounts themselves. By default, this checks all accounts marked as 'error' to

determine if accounts are still in an error state (i.e. disabled or locked). If you specify the 'AllAccounts' option, it checks all accounts regardless of error state and updates their status.

Examples

EXAMPLE 1

Checks the status of accounts in the identity pool MyPool that are currently in the Error state, marking them as Available, InUse, Tainted or Error as appropriate.

```
1 Update-AcctADAccount -IdentityPoolName MyPool
```

EXAMPLE 2

Checks the status of all accounts in the identity pool MyPool marking them as Available, InUse, Tainted or Error as appropriate.

```
1 Update-AcctADAccount -IdentityPoolName MyPool -AllAccounts
```

Parameters

-IdentityPoolName

The name of the identity pool of the AD accounts that are to be updated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier for the identity pool of the AD accounts that are to be updated.

Type:	Guid

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AllAccounts

Indicates if all accounts should be updated or only the ones marked as in error.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the Citrix Virtual Apps and Desktops logs.

Related Links

- about_AcctADIdentitySnapin
- New-AcctADAccount
- Add-AcctADAccount
- Remove-AcctADAccount
- Repair-AcctADAccount
- Unlock-AcctADAccount
- Get-AcctADAccount

about_AnalyticsAnalyticsSnapIn

March 11, 2024

Topic

about_AnalyticsAnalyticsSnapIn

Short Description

The Analytics PowerShall snap-in provides administrative functions for the Analytics Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Analytics'.

Long Description

The Analytics Service PowerShell snap-in enables both local and remote administration of the Analytics Service.

The Analytics Service collects Citrix CEIP data about the DDC. For more information about CEIP, see http://more.citrix.com/XD-CEIP.

This snap-in provides commands for a Citrix administrator to enable a data collection, import the collection data points file, and query the current status of the Analytics database configuration and service.

For more information on the Analytics Service, see the help top about Analytics Service.

Use the command 'Get-Command -Module Citrix.Analytics.Admin.V1'to see a complete list of commands supported by this snap-in.

For usage of each command, use the Get-Help command to display the help text for each command.

In addition to pre-programed schedules hard-coded in the Analytics Service, administrators can use the Set-AnalyticsSite command to trigger immediate data collection and upload. The registry string value HKLM/Software/Citrix/XDServices/Analytics/DoOperation can be specified to indicate the type of collection to be performed upon the execution of the Set-AnalyticsSite command. If the data stored in this registry value is set to 'Collect', then upon issuance of the command 'Set-AnalyticsSite -Enabled $true', the Analytics Service starts a collection immediately. The result of the collection is a .zip file, whose location is stored in the the registry value 'DoOperationResult'. The collected result is not uploaded to the Citrix CEIP site.

If the value of 'DoOperation'is 'CollectAndUpload', then a collection is performed and the result is also uploaded to the Citrix CEIP site.

If the value of 'DoOperation'is 'CollectSite', then only site data is collected. For detailed list of site data, refer to the 'Scope'parameter for each data point in the DataPoints.xml file, which can be found under C:\Program Files\Citrix\XenDesktopPoshSDK\Module\Citrix.XenDesktop.Admin.V1\Citrix.XenDesktop.Admin.

If the value of 'DoOperation'is 'CollectHost', then only host data is collected, as specified in the data points file.

The data points file can be imported using the Import-AnalyticsDataDefinition file. The file must be a file signed by Citrix.

When running, the Analytics Service periodically collects data using the latest data point file stored in the DDC database. The collection runs at approximately every 30 minutes to perform data collection. Each collection runs for 5 to 10 minutes, depending on the size of data to be collected. The service has been optimized so that minimal system resources are used to do the data collection.

If collection results are not uploaded immediately, they can be manually uploaded. If they are not manually uploaded, they will be periodically packaged and uploaded by the Analytics Service. If for any reason an upload fails, the Service will try again but not before doubling the amount of time between the most recent to successful uploads. If upload keeps failing, very soon the Service will wait a very long time to attempt the next upload. The time can be weeks or months. A system reboot resets the upload wait time.

The collection and upload algorithm and schedules are hard-coded in the Service and cannot be changed.

about_AnalyticsAnalyticsSnapIn

March 11, 2024

Topic

about_AnalyticsAnalyticsSnapIn

Short Description

The Analytics PowerShall snap-in provides administrative functions for the Analytics Service.

Command Prefix

All commands in this snap-in have the noun prefixed with ‘Analytics’.

Long Description

The Analytics Service PowerShell snap-in enables both local and remote administration of the Analytics Service.

The Analytics Service collects Citrix CEIP data about the DDC. For more information about CEIP, see http://more.citrix.com/XD-CEIP.

This snap-in provides commands for a Citrix administrator to enable a data collection, import the collection data points file, and query the current status of the Analytics database configuration and service.

For more information on the Analytics Service, see the help top about Analytics Service.

Use the command 'Get-Command -Module Citrix.Analytics.Admin.V1'to see a complete list of commands supported by this snap-in.

For usage of each command, use the Get-Help command to display the help text for each command.

In addition to pre-programed schedules hard-coded in the Analytics Service, administrators can use the Set-AnalyticsSite command to trigger immediate data collection and upload. The registry string value HKLM/Software/Citrix/XDServices/Analytics/DoOperation can be specified to indicate the type of collection to be performed upon the execution of the Set-AnalyticsSite command. If the data stored in this registry value is set to 'Collect', then upon issuance of the command 'Set-AnalyticsSite -Enabled $true', the Analytics Service starts a collection immediately. The result of the collection is a .zip file, whose location is stored in the the registry value 'DoOperationResult'. The collected result is not uploaded to the Citrix CEIP site.

If the value of 'DoOperation'is 'CollectAndUpload', then a collection is performed and the result is also uploaded to the Citrix CEIP site.

If the value of 'DoOperation'is 'CollectSite', then only site data is collected. For detailed list of site data, refer to the 'Scope'parameter for each data point in the DataPoints.xml file, which can be found under C:\Program Files\Citrix\XenDesktopPoshSDK\Module\Citrix.XenDesktop.Admin.V1\Citrix.XenDesktop.Admin.

If the value of 'DoOperation'is 'CollectHost', then only host data is collected, as specified in the data points file.

The data points file can be imported using the Import-AnalyticsDataDefinition file. The file must be a file signed by Citrix.

When running, the Analytics Service periodically collects data using the latest data point file stored in the DDC database. The collection runs at approximately every 30 minutes to perform data collection. Each collection runs for 5 to 10 minutes, depending on the size of data to be collected. The service has been optimized so that minimal system resources are used to do the data collection.

If collection results are not uploaded immediately, they can be manually uploaded. If they are not manually uploaded, they will be periodically packaged and uploaded by the Analytics Service. If for any reason an upload fails, the Service will try again but not before doubling the amount of time between the most recent to successful uploads. If upload keeps failing, very soon the Service will wait a very long time to attempt the next upload. The time can be weeks or months. A system reboot resets the upload wait time.

The collection and upload algorithm and schedules are hard-coded in the Service and cannot be changed.

about_Analytics_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or

descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like ‘‘High*’’-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company ‘‘citrix’’-Product ‘[X]EN*’
Get-<Noun> -Product ‘‘Xen*’’-Company ‘‘CITRIX’’
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:


```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago
```

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}
```


You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{
```



```
6
7    $sessions = @(Get-BrokerSession -Filter {
8    Uid -gt $lastUid  }
9    -MaxRecordCount 1000 -Sortby 'Uid')
```

```
if ($sessions.Length -eq 0)
{
break;
}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Get-AnalyticsDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Analytics Service.

Syntax

```
Get-AnalyticsDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected Analytics Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Analytics Service instance running on controller "controller1.mydomain.net".

```
Get-AnalyticsDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Analytics Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the AnalyticsService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Set-AnalyticsDBConnection
- Get-AnalyticsServiceStatus
- Test-AnalyticsDBConnection

Get-AnalyticsDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Analytics Service.

Syntax

```
Get-AnalyticsDBSchema
   [-DatabaseName <String>]
   [-ServiceGroupName <String>]
   [-ScriptType <ScriptTypes>]
   [-LocalDatabase]
   [-Sid <String>]
   [-DatabaseRights <String>]
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets SQL scripts that can be used to create a new Citrix Analytics Service database schema, add a new Analytics service to an existing site, remove a Analytics service from a site, or create a database server logon for a Analytics service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Analytics service instance, otherwise the scripts relate to Analytics service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Analytics service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Analytics service roles

If ScriptType is Evict, the returned script contains:

- Removal of Analytics service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Analytics Service and copies it to a file called "C:\AnalyticsSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Analytics service schema.

```
Get-AnalyticsDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\AnalyticsSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Analytics service. This can be used when configuring a mirror server for use.

```
Get-AnalyticsDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    AnalyticsLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Analytics service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Analytics services that share the same database instance and are con-

sidered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Analytics Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Analytics Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Analytics Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Analytics Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Analytics services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Analytics Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Set-AnalyticsDBConnection
- Test-AnalyticsDBConnection

Get-AnalyticsDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Analytics Service.

Syntax

```
Get-AnalyticsDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Analytics Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Analytics Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Analytics Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Analytics Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Analytics Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-AnalyticsServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-AnalyticsDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Analytics Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-AnalyticsDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Analytics Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shut down during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Analytics services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AnalyticsServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsInstalledDBVersion
- Get-AnalyticsServiceStatus
- Get-AnalyticsDBSchema

Get-AnalyticsInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Analytics Service.

Syntax

```
Get-AnalyticsInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Analytics Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-AnalyticsDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Analytics Service database schema version number.

```
Get-AnalyticsInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Analytics Service database schema for which upgrade scripts are supplied.

```
Get-AnalyticsInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-AnalyticsInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Analytics

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsDBVersionChangeScript
- Get-AnalyticsDBSchema

Get-AnalyticsService

March 11, 2024

Gets the service record entries for the Analytics Service.

Syntax

```
Get-AnalyticsService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the Analytics Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Analytics Service running in the current service group.

```
Get-AnalyticsService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```


Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*"matches records with a metadata entry having a key name of "abc"and a value starting with the letter "x".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Analytics_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Analytics_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Analytics.Sdk.Service

The Get-AnalyticsServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin

Get-AnalyticsServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Analytics Service on the controller.

Syntax

```
Get-AnalyticsServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Analytics Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Analytics Service.

```
Get-AnalyticsServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin

Get-AnalyticsServiceInstance

March 11, 2024

Gets the service instance entries for the Analytics Service.

Syntax

```
Get-AnalyticsServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Analytics Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Analytics Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-AnalyticsServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Analytics.Sdk.ServiceInstance

The Get-AnalyticsServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Analytics.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Analytics.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsServiceStatus
- Reset-AnalyticsServiceGroupMembership

Get-AnalyticsServiceStatus

March 11, 2024

Gets the current state of the Analytics Service on the controller.

Syntax

```
Get-AnalyticsServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the Analytics Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will

return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Analytics Service.

```
Get-AnalyticsServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-AnalyticsServiceStatus command returns an object containing the status of the Analytics Service together with extra diagnostics information.

DBUnconfigured

The Analytics Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Analytics Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Analytics Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Analytics is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Analytics is connected to a database that is valid, but it does not have the full functionality required so the Analytics cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Analytics Service currently in use is incompatible with the version of the Analytics Service schema on the database. Upgrade the Analytics Service to a more recent version.

DBOlderVersionThanService

The version of the Analytics Service schema on the database is incompatible with the version of the Analytics Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Analytics Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Analytics Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Analytics and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Analytics service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Set-AnalyticsDBConnection
- Test-AnalyticsDBConnection
- Get-AnalyticsDBConnection
- Get-AnalyticsDBSchema

Get-AnalyticsSite

March 11, 2024

This cmdlet returns the site configuration for the Analytics Service/Customer Experience Improvement Program. Details are available at http://more.citrix.com/XD-CEIP

Syntax

```
Get-AnalyticsSite
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The configuration will be contained in the Site object that is returned. The Site object has two properties, Enabled, and DataDefinitionVersion. If Enabled is True, then site is participating in the Citrix Experience Improvement Program.

Examples

EXAMPLE 1

This retrieves the Analytics Site configuration.

```
Get-AnalyticsSite

DataDefinitionVersion                          Enabled
---------------------                          -------
1.0.0                                          False
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Analytics.Sdk.Site

The site object with configuration properties.

Related Links

- about_AnalyticsAnalyticsSnapin

Import-AnalyticsDataDefinition

March 11, 2024

This imports a new data definition file for the Analytics Service/Customer Experience Improvement Program.

Syntax

```
Import-AnalyticsDataDefinition
      -DataDefinition <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

The DataDefinition file defines what data should be collected and sent to Citrix. Only Citrix signed DataDefinition files can be imported. An error occurs if the file is not signed properly.

Examples

EXAMPLE 1

This imports a new DataDefinition file.

```
1 Import-AnalyticsDataDefinition -DataDefinition c:\SignedDataDefinition.
    xml
```

Parameters

-DataDefinition

The path to a new DataDefinition file

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_AnalyticsAnalyticsSnapin

Remove-AnalyticsServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-AnalyticsServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AnalyticsServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AnalyticsServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AnalyticsServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-AnalyticsService | % {
  Remove-AnalyticsServiceMetadata -Map $_.MetadataMap  }
```


Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Set-AnalyticsServiceMetadata

Reset-AnalyticsEnabledFeatureList

March 11, 2024

Refreshes the Analytics service's list of enabled features.

Syntax

```
Reset-AnalyticsEnabledFeatureList
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Analytics Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Analytics service instance's list of enabled features.

```
1  Reset-AnalyticsEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-AnalyticsServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Analytics Service.

Syntax

```
Reset-AnalyticsServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Analytics Service access permissions and configuration service locations. The Reset-AnalyticsServiceGroupMembership command must be run on at least one instance of the service type (Analytics) after installation and registration with the configuration service. Without this operation, the Analytics services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AnalyticsServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    AnalyticsServiceGroupMembership
```


EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-AnalyticsServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Analytics.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AnalyticsServiceGroupMembership command.

Outputs

Citrix.Analytics.Sdk.ServiceInstance

Reset-AnalyticsServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Analytics Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsServiceInstance
- Get-AnalyticsServiceStatus

Set-AnalyticsDBConnection

March 11, 2024

Configures a database connection for the Analytics Service.

Syntax

```
Set-AnalyticsDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Analytics Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-AnalyticsDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-AnalyticsDBConnection -DBConnection $null
```


Parameters

-DBConnection

Specifies the database connection string to be used by the Analytics Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-AnalyticsDBConnection cmdlet returns an object describing the status of the Analytics Service together with extra diagnostics information. Possible values are:

- OK:

The Analytics Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Analytics Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Analytics Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Analytics Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Analytics Service currently in use is newer than, and incompatible with, the version of the Analytics Service schema on the database. Upgrade the Analytics Service to a more recent version.

- DBOlderVersionThanService:

The version of the Analytics Service schema on the database is newer than, and incompatible with, the version of the Analytics Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Analytics Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Analytics Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Analytics Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsServiceStatus
- Get-AnalyticsDBConnection
- Test-AnalyticsDBConnection

Set-AnalyticsDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Analytics Service.

Syntax

```
Set-AnalyticsDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AnalyticsDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Analytics Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Analytics Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-AnalyticsDBCredentials
```


EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-AnalyticsDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-AnalyticsDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-AnalyticsDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AnalyticsDBSchema
- Set-AnalyticsDBConnection
- Get-Credential

Set-AnalyticsServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-AnalyticsServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AnalyticsServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AnalyticsServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AnalyticsServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.


```
Set-AnalyticsServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AnalyticsServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Remove-AnalyticsServiceMetadata

Set-AnalyticsSite

March 11, 2024

This sets the site configuration for the Analytics Service/Customer Experience Improvement Program. Details are available at http://more.citrix.com/XD-CEIP

Syntax

```
Set-AnalyticsSite
    [-Enabled <Boolean>]
    [-NewSite <Site>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this cmdlet to opt-in to the Citrix Experience Improvement Program. For advanced users it can also take a new Citrix.Analytics.Sdk.Site object using the -NewSite parameter.

Examples

EXAMPLE 1

This enables the Citrix Experience Improvement Program. To disable use $false instead.

```
Set-AnalyticsSite -Enabled $true
```


Parameters

-Enabled

This enables or disables the Citrix Experience Improvement Program functionality.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NewSite

New site object, if you wish to supply the a whole Site object.

Type:	Site
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_AnalyticsAnalyticsSnapin

Test-AnalyticsDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Analytics Service.

Syntax

```
Test-AnalyticsDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AnalyticsDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AnalyticsDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Analytics Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Analytics SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.


```
Test-AnalyticsDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Analytics Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-AnalyticsDBConnection cmdlet returns an object describing the status of the selected Analytics Service instance that would result if the connection string were used with the Set-AnalyticsDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-AnalyticsDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Analytics Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Analytics Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Analytics Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Analytics Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Analytics Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Analytics Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AnalyticsAnalyticsSnapin
- Get-AnalyticsServiceStatus
- Get-AnalyticsDBConnection
- Set-AnalyticsDBConnection

about_AppLibAppLibrarySnapin

March 11, 2024

Topic

about_AppLibAppLibrarySnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'AppLib'.

Long Description

This service long description

about_AppLibAppLibrarySnapin

March 11, 2024

Topic

about_AppLibAppLibrarySnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'AppLib'.

Long Description

This service long description

about_AppLib_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                  # Matches Abc*
Get-<Noun> -Company "Abc`*"                   # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```


Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for

objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings

formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-AppLibIsolationGroupPackage

March 11, 2024

Adds an App-V package to an Isolation Group.

Syntax

```
Add-AppLibIsolationGroupPackage
   [-IsolationGroupUid] <Int32>
   -AppVPackageUid <Int32>
   -OrderNumber <Int32>
   [-ExplicitInclusion <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Adds the App-V package to the specified Isolation Group.

Examples

EXAMPLE 1

Adds the App-V package with the unique identifier to the specified isolation group.

```
Add-AppLibIsolationGroupPackage -Uid 4 -IsolationGroupUid 15 -
   ExplicitInclusion $true -OrderNumber 1
```


Parameters

-IsolationGroupUid

The AppLibrary's internal unique identifier of the Isolation Group to contain the package.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppVPackageUid

The AppLibrary's internal unique identifier of the App-V package.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OrderNumber

The order number within the Isolation Group.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-ExplicitInclusion

Indicates whether the package is explicitly included in the delivery group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.IsolationGroupPackage

An object representing the IsolationGroupPackage.

Related Links

Get-AppLibAppAttachApplicationsPolicy

March 11, 2024

Gets the updated policy for the list of applications passed

Syntax

```
Get-AppLibAppAttachApplicationsPolicy
    [[-AppIdentifierList] <String[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet should be used to get policy for published app attach applications. The input should be a list of app attach application identifiers list

Examples

EXAMPLE 1

Gets the updated policy for the list of applications passed

```
$appIdentifiers = "4345-fr23320-df","45234-sdewr-gfgf","785ddf-vcvxdsf-dfdf"
Get-AppLibAppAttachApplicationsPolicy -AppIdentifierList $appIdentifiers
```

Parameters

-AppIdentifierList

List of App attach application identifiers

Type:	String[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

byte array containing the updated policy

Notes

Pass in the App Identifiers as a comma separated string

Related Links

Get-AppLibAppVApplication

March 11, 2024

Gets the details of an App-V application held in the application library.

Syntax

```
Get-AppLibAppVApplication
    [[-Uid] <Int32>]
    [-Name <String>]
    [-AppVPackageUid <Int32>]
    [-Description <String>]
    [-RetrieveIcon <Boolean>]
    [-RetrievePolicy <Boolean>]
    [-RetrieveUsers <Boolean>]
    [-RetrieveMetadata <Boolean>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The application library holds the information required to find and launch an App-V application on a client VDA.

Examples

EXAMPLE 1

Gets the details for all of the App-V applications in the library.

```
Get-AppLibAppVApplication
```

EXAMPLE 2

Gets the details of the App-V application named MyApp.

```
Get-AppLibAppVApplication -Name "MyApp"
```

EXAMPLE 3

Gets the details of all of the App-V applications in the specified package.


```
Get-AppLibAppVApplication -AppVPackageUid 15
```

EXAMPLE 4

Gets the details of all App-V applications without the metadata

```
Get-AppLibAppVApplication -RetrieveMetadata $false
```

Parameters

-Uid

The AppLibrary's internal UID of the App-V application.

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

The name of the application in the the App-V package.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AppVPackageUid

The AppLibrary's internal UID of the App-V package that contains the application.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description of the application.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RetrieveIcon

A switch to indicate whether to return the application's icon data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RetrievePolicy

A switch to indicate whether to return the application's policy blob

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RetrieveUsers

A switch to indicate whether to return the application's associated users

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RetrieveMetadata

A switch to indicate whether to return the application's metadata

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.AppVApplication

An object representing the details required to launch the App-V application.

Notes

Pass in the App-V application's UID to retrieve a single application.

Pass in a package UID to retrieve all of the applications in that package.

Call the cmdlet without any parameters to retrieve all of the applications in the library.

Related Links

Get-AppLibAppVApplicationInfo

March 11, 2024

Enumerates information for a given application in a given package

Syntax

```
Get-AppLibAppVApplicationInfo
   -PackageGuid <String>
   -ApplicationId <Int32>
   [-LibraryId <Int32>]
   [-Property <String[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to query application information about single admin appv package. Currently only fetching filetype associations is supported. See example

Examples

EXAMPLE 1

Fetches all the file type associations for the given application

```
Get-AppLibAppVApplicationInfo -PackageGuid "7796B8F3-713A-446F-840D-18
    E5171AB21A" -ApplicationId 14
```

Parameters

-PackageGuid

The package Guid

Type:	String
Aliases:	packageId
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ApplicationId

The AppV application Id

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LibraryId

The Application Library Id

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Property

The property parameter is reserved for future use. Currently only FileTypeAssociations are supported.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Object with required information wihtin object as properties

Notes

Currently only fetching file type associations are supported.

Related Links

Get-AppLibAppVApplicationsPolicy

March 11, 2024

Gets the updated policy for the list of applications passed

Syntax

```
Get-AppLibAppVApplicationsPolicy
    [[-AppIdentifierList] <String[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet should be used to get policy for published single admin App-V applications. The input should be a list of App-V application identifiers list

Examples

EXAMPLE 1

Gets the updated policy for the list of applications passed

```
$appIdentifiers = "4345-fr23320-df","45234-sdewr-gfgf","785ddf-vcvxdsf-
    dfdf"
Get-AppLibAppVApplicationsPolicy -AppIdentifierList $appIdentifiers
```

Parameters

-AppIdentifierList

List of App-V application identifiers

Type:	String[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

byte array containing the updated policy

Notes

Pass in the App Identifiers as a comma separated string

Related Links

Get-AppLibAppVPackage

March 11, 2024

Gets the details of an App-V package held in the application library.

Syntax

```
Get-AppLibAppVPackage
   [[-Uid] <Int32>]
   [-Name <String>]
   [-LibraryUid <Int32>]
   [-Description <String>]
   [-RetrieveIcon <Boolean>]
   [-RetrieveMetadata <Boolean>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-AppLibAppVPackage
   [[-Uid] <Int32>]
   [-Name <String>]
   [-LibraryUid <Int32>]
   [-Description <String>]
   [-RetrieveIcon <Boolean>]
   -RetrieveApplicationPolicy <Boolean>
   -RetrieveApplicationData <Boolean>
   [-RetrieveMetadata <Boolean>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
```



```
17      [<CitrixCommonParameters>]
18      [<CommonParameters>]
```

Description

The application library holds the information about App-V packages, their location on the network, and the applications they contain.

Examples

EXAMPLE 1

Gets the details of all of the App-V packages in the AppLibrary.

```
1  Get-AppLibAppVPackage
```

EXAMPLE 2

Gets the details of the App-V package named MyPackage.

```
1  Get-AppLibAppVPackage -Name "MyPackage"
```

EXAMPLE 3

Gets the details of all of the App-V packages in the specified library.

```
1  Get-AppLibAppVPackage -libraryUid 1
```

EXAMPLE 4

Gets the details of the App-V package named MyPackage, including it and its containing application's icon data.

```
1  Get-AppLibAppVPackage -Name "MyPackage" -RetrieveIcon $true
```

EXAMPLE 5

Gets the details of the App-V package named MyPackage, including it and its containing application and their policy data.


```
Get-AppLibAppVPackage -Name "MyPackage" -RetrieveApplicationData $true
    -RetrieveApplicationPolicy $true
```

EXAMPLE 6

Gets the details of the App-V package named MyPackage. Here we are setting the retrieveMedata flag to false, so that package info will be returned without metadata

```
Get-AppLibAppVPackage -Name "MyPackage" -RetrieveMetadata $false
```

Parameters

-RetrieveApplicationPolicy

A switch to indicate whether to return the package application's policy data data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-RetrieveApplicationData

A switch to indicate whether to return the package applications.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Uid

The AppLibrary's internal UID of the App-V package.

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

The name of the App-V package.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LibraryUid

The AppLibrary's internal UID of the specific library that holds details of the App-V package.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description of the App-V package and its contents.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RetrieveIcon

A switch to indicate whether to return the package's icon data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RetrieveMetadata

A switch to indicate whether to return the package metadata

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.AppVPackage

An object representing the details needed to identify an App-V package.

Notes

Pass in the App-V package's UID to retrieve a single App-V package.

Pass in a library UID to retrieve all af the packages in that library.

Call the cmdlet without any parameters to retrieve all of the packages in the AppLibrary.

Related Links

Get-AppLibAppVServer

March 11, 2024

Gets the details of an App-V Server created in the application library.

Syntax

```
Get-AppLibAppVServer
    [-Uid <Int32>]
    [-ManagementServer <String>]
    [-PublishingServer <String>]
    [-RetrievePolicy <Boolean>]
    [-RetrievePackageIds <Boolean>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The application library holds the information about App-V Servers and their associated policies

Examples

EXAMPLE 1

Gets the details of all of the App-V servers in the AppLibrary.

```
Get-AppLibAppVServer
```

EXAMPLE 2

Gets the details of the App-V server with Uid 1.

```
Get-AppLibAppVServer -Uid 1
```


EXAMPLE 3

Gets the details of the App-V server with Uid 1, including it and its containing policy data.

```
Get-AppLibAppVServer -Uid 1 -RetrievePolicy $true
```

EXAMPLE 4

Gets the details of the App-V server with given Management Server and Publishing server urls, including it and its containing policy data.

```
Get-AppLibAppVServer -ManagementServer http://AppVSrv.company.local -
    PublishingServer http://AppVSrv.company.local:8001 -RetrievePolicy
    $true
```

Parameters

-Uid

Uid of the App-V server.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ManagementServer

Management Server Url of the App-V server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	True

-PublishingServer

Publishing Server Url of the App-V server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-RetrievePolicy

A switch to indicate whether to return the servers policy data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-RetrievePackageIds

A switch to indicate whether to return the server package Ids.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.AppVServer

An object representing the details needed to identify an App-V server.

Notes

Pass in the App-V server's UID to retrieve a single App-V server.

Call the cmdlet without any parameters to retrieve all of the servers in the AppLibrary.

Related Links

Get-AppLibDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the AppLibrary Service.

Syntax

```
Get-AppLibDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected AppLibrary Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the AppLibrary Service instance running on controller "controller1.mydomain.net".

```
Get-AppLibDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current AppLibrary Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the AppLibraryService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibDBConnection
- Get-AppLibServiceStatus
- Test-AppLibDBConnection

Get-AppLibDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix AppLibrary Service.

Syntax

```
Get-AppLibDBSchema
   [-DatabaseName <String>]
   [-ServiceGroupName <String>]
   [-ScriptType <ScriptTypes>]
   [-LocalDatabase]
   [-Sid <String>]
   [-DatabaseRights <String>]
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets SQL scripts that can be used to create a new Citrix AppLibrary Service database schema, add a new AppLibrary service to an existing site, remove a AppLibrary service from a site, or create a database server logon for a AppLibrary service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected AppLibrary service instance, otherwise the scripts relate to AppLibrary service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to AppLibrary service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to AppLibrary service roles

If ScriptType is Evict, the returned script contains:

- Removal of AppLibrary service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix AppLibrary Service and copies it to a file called "C:\AppLibrarySchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a AppLibrary service schema.

```
Get-AppLibDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\AppLibrarySchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the AppLibrary service. This can be used when configuring a mirror server for use.

```
Get-AppLibDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    AppLibraryLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new AppLibrary service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the AppLibrary services that share the same database instance and are

considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix AppLibrary Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a AppLibrary Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a AppLibrary Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the AppLibrary Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for AppLibrary services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the AppLibrary Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibDBConnection
- Test-AppLibDBConnection

Get-AppLibDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix AppLibrary Service.

Syntax

```
Get-AppLibDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix AppLibrary Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix AppLibrary Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix AppLibrary Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix AppLibrary Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix AppLibrary Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-AppLibServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-AppLibDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix AppLibrary Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-AppLibDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix AppLibrary Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any AppLibrary services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AppLibServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibInstalledDBVersion
- Get-AppLibServiceStatus
- Get-AppLibDBSchema

Get-AppLibFlexAppApplicationsPolicy

March 11, 2024

Gets the updated policy for the list of applications passed

Syntax

```
Get-AppLibFlexAppApplicationsPolicy
    [[-AppIdentifierList] <String[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet should be used to get policy for published FlexApp applications. The input should be a list of FlexApp application identifiers list

Examples

EXAMPLE 1

Gets the updated policy for the list of applications passed

```
$appIdentifiers = "ac4c2abf-457b-43e2-9141-432546b13d7a","eb95c4ab-aa01
    -495e-aa19-13b435d0fd54","edead135-abcd-4b1b-8dc6-f46b402695c5"
Get-AppLibFlexAppApplicationsPolicy -AppIdentifierList $appIdentifiers
```


Parameters

-AppIdentifierList

List of FlexApp application identifiers

Type:	String[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

byte array containing the updated policy

Notes

Pass in the App Identifiers as a comma separated string

Related Links

Get-AppLibInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the AppLibrary Service.

Syntax

```
Get-AppLibInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix AppLibrary Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-AppLibDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix AppLibrary Service database schema version number.

```
1 Get-AppLibInstalledDBVersion
```

EXAMPLE 2

Get the versions of the AppLibrary Service database schema for which upgrade scripts are supplied.

```
1 Get-AppLibInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-AppLibInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the AppLibrary

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibDBVersionChangeScript
- Get-AppLibDBSchema

Get-AppLibIsolationGroup

March 11, 2024

Gets the details of an isolation group held in the application library.

Syntax

```
Get-AppLibIsolationGroup
   [-IncludePolicy <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-AppLibIsolationGroup
   [[-IsolationGroupUid] <Int32>]
   [-IncludePolicy <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-AppLibIsolationGroup
   [[-Name] <String>]
   [-IncludePolicy <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The application library holds the information about isolation groups, their location on the network and the packages they contain.

Examples

EXAMPLE 1

Gets the details for all of the isolation groups in the AppLibrary.

```
Get-AppLibIsolationGroup
```

EXAMPLE 2

Gets the details of the isolation group named MyIsolationGroup.

```
Get-AppLibIsolationGroup -Name "MyIsolationGroup"
```

EXAMPLE 3

Gets the details for the isolation with a Uid of 5.

```
Get-AppLibIsolationGroup -IsolationGroupUid 5
```

EXAMPLE 4

Gets the details for the isolation with a Uid of 5 and also returns the Citrix Group Policy data that defines the object.

```
Get-AppLibIsolationGroup -IsolationGroupUid 5 -IncludePolicy $true
```

Parameters

-IsolationGroupUid

The App Library's internal unique identifier of the isolation group.

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-Name

The name of the isolation group to fetch.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-IncludePolicy

The option to retrieve the Isolation Group policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.IsolationGroup

An object representing the details needed to identify an isolation group.

Notes

Pass in the isolation group's uid to retreive a single isolation group.

Pass in a library uid to retrieve all of the isolation groups in that library.

Call the cmdlet without any parameters to retieve all of the isolation groups in the AppLibrary.

Related Links

Get-AppLibIsolationGroupPackage

March 11, 2024

Gets the details of an App-V package of an Isolation Group.

Syntax

```
Get-AppLibIsolationGroupPackage
    [[-IsolationGroupUid] <Int32>]
    [-AppVPackageUid <Int32>]
    [-ExplicitInclusion <Boolean>]
    [-OrderNumber <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves the App-V package of the specified Isolation Group.

Examples

EXAMPLE 1

Gets the details of the App-V package with the unique identifier in the specified isolation group.

```
Get-AppLibIsolationGroupPackage -AppVPackageUid 4 -IsolationGroupUid 15
```

EXAMPLE 2

Gets the details for all of the App-V packages in the specified isolation group.

```
Get-AppLibIsolationGroupPackage -IsolationGroupUid 15
```


Parameters

-IsolationGroupUid

The AppLibrary's internal unique identifier of the Isolation Group which contains the package.

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-AppVPackageUid

The AppLibrary's internal unique identifier of the App-V package.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExplicitInclusion

Indicates whether the package is explicitly included in the delivery group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-OrderNumber

The order number within the Isolation Group (if the package belongs to an Isolation Group).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or

descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.IsolationGroupPackage

An object representing the details of an App-V package in an Isolation Group.

Related Links

Get-AppLibLibrary

March 11, 2024

Gets the details of a library maintained by the AppLibrary service.

Syntax

```
Get-AppLibLibrary
   [-Uid <Int32>]
   [[-Name] <String>]
   [-Description <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
```



```
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The AppLibrary service maintains information about various libraries. Network locations that are monitored for changes to the App-V packages are stored there.

Examples

EXAMPLE 1

Gets the details of all of the libraries maintained by the AppLibrary service.

```
Get-AppLibLibrary
```

EXAMPLE 2

Gets the details of the library named Manual.

```
Get-AppLibLibrary -Name "Manual"
```

EXAMPLE 3

Gets the details of the specified library.

```
Get-AppLibLibrary -Uid 1
```

Parameters

-Name

The name of the application library.

Type: String

Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Uid

The AppLibrary's internal UID of the library.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description of the application library.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.Library

An object representing the details required to maintain a library of App-V applications.

Notes

Pass in a library UID to retrieve only that specific library.

Call the cmdlet without any parameters to retrieve all of the libraries maintained by the AppLibrary service.

The 'Manual'library is not actively monitored and is designed to be managed by the user for ad-hoc storage of packages in unstructured locations.

Related Links

Get-AppLibMsixApplicationsPolicy

March 11, 2024

Gets the updated policy for the list of applications passed

Syntax

```
Get-AppLibMsixApplicationsPolicy
   [[-AppIdentifierList] <String[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet should be used to get policy for published Msix applications. The input should be a list of Msix application identifiers list

Examples

EXAMPLE 1

Gets the updated policy for the list of applications passed

```
$appIdentifiers = "ac4c2abf-457b-43e2-9141-432546b13d7a","eb95c4ab-aa01
    -495e-aa19-13b435d0fd54","edead135-abcd-4b1b-8dc6-f46b402695c5"
Get-AppLibMsixApplicationsPolicy -AppIdentifierList $appIdentifiers
```

Parameters

-AppIdentifierList

List of Msix application identifiers

Type:	String[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

byte array containing the updated policy

Notes

Pass in the App Identifiers as a comma separated string

Related Links

Get-AppLibPackageDiscovery

March 11, 2024

Gets information about the package discovery sessions that are currently being run by the AppLibrary Service

Syntax

```
Get-AppLibPackageDiscovery
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-AppLibPackageDiscovery
   -Id <Guid>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-AppLibPackageDiscovery
   -DiscoveryProfileUid <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

A package discovery object has information about the state and progress of the discovery session

Examples

EXAMPLE 1

Gets information about all of the package discovery sessions that are being managed by the AppLibrary service

```
Get-AppLibPackageDiscovery
```

EXAMPLE 2

Gets information about a particular package discovery session which matches the specified Id

```
Get-AppLibPackageDiscovery -Id 1
```

Parameters

-Id

The unique identifier of a running (or recently completed) package discovery session

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoveryProfileUid

The package discovery profile to filter the discovery sessions with

Type:	Int32
Position:	Named
Default value:	0
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

Unique identifier

Outputs

Citrix.AppLibrary.Sdk.PackageDiscoveryProfile

Information about the state of the package discovery session

Related Links

- New-AppLibPackageDiscoveryProfile
- Get-AppLibPackageDiscoveryProfile
- Set-AppLibPackageDiscoveryProfile
- Remove-AppLibPackageDiscoveryProfile
- New-AppLibPackageDiscovery

Get-AppLibPackageDiscoveryProfile

March 11, 2024

Gets the package discovery profiles that are defined in the AppLibrary

Syntax

```
Get-AppLibPackageDiscoveryProfile
    [-Uid <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

A package discovery can be used by the AppLibrary Service to automatically (or periodically) search for application packages to import into the AppLibrary

Examples

EXAMPLE 1

Gets all the package discovery profiles that are defined in the AppLibrary Service

```
Get-AppLibPackageDiscoveryProfile
```

EXAMPLE 2

Gets a particular package discovery profile which matches the specified Uid

```
Get-AppLibPackageDiscoveryProfile -Uid 1
```

Parameters

-Uid

The unique identifier of a particular package discovery profile to return

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Int32

Numerical data

Outputs

Citrix.AppLibrary.Sdk.PackageDiscoveryProfile

Information about how, where and when the package discovery will be run

Related Links

- New-AppLibPackageDiscoveryProfile
- Set-AppLibPackageDiscoveryProfile
- Remove-AppLibPackageDiscoveryProfile
- New-AppLibPackageDiscovery
- Get-AppLibPackageDiscovery

Get-AppLibService

March 11, 2024

Gets the service record entries for the AppLibrary Service.

Syntax

```
Get-AppLibService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the AppLibrary Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the AppLibrary Service running in the current service group.

```
Get-AppLibService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.Service

The Get-AppLibServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin

Get-AppLibServiceAddedCapability

March 11, 2024

Gets any added capabilities for the AppLibrary Service on the controller.

Syntax

```
Get-AppLibServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the AppLibrary Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the AppLibrary Service.

```
Get-AppLibServiceAddedCapability
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin

Get-AppLibServiceConfigurationData

March 11, 2024

Gets configuration data for the service.

Syntax

```
Get-AppLibServiceConfigurationData
    [-Name <String[]>]
    [-Value <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Provides the ability to determine the configuration parameters for the service

Examples

EXAMPLE 1

Get the Configuration data for the service.

```
Get-AppLibConfigurationData

Name                : DeltaDiskDelete.timeDelay
Value               : 10
```

Parameters

-Name

The Configuration data item Name .

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Value

The Configuration data item value.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_AppLib_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32

Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_AppLib_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.ServiceConfigurationData

This object provides details of the configuration data and contains the following information:

Name <string>

The Name for the configuration item.

Value <string>

The value for the configuration item.

Notes

In the case of failure the following errors can be produced.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibServiceConfigurationData
- Remove-AppLibServiceConfigurationData

Get-AppLibServiceInstance

March 11, 2024

Gets the service instance entries for the AppLibrary Service.

Syntax

```
Get-AppLibServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the AppLibrary Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the AppLibrary Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-AppLibServiceInstance
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.ServiceInstance

The Get-AppLibServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always AppLib.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.AppLibrary.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibServiceStatus
- Reset-AppLibServiceGroupMembership

Get-AppLibServiceStatus

March 11, 2024

Gets the current state of the AppLibrary Service on the controller.

Syntax

```
Get-AppLibServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Enables the status of the AppLibrary Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the AppLibrary Service.

```
1  Get-AppLibServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-AppLibServiceStatus command returns an object containing the status of the AppLibrary Service together with extra diagnostics information.

DBUnconfigured

The AppLibrary Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the AppLibrary Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the AppLibrary Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The AppLibrary is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The AppLibrary is connected to a database that is valid, but it does not have the full functionality required so the AppLibrary cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the AppLibrary Service currently in use is incompatible with the version of the AppLibrary Service schema on the database. Upgrade the AppLibrary Service to a more recent version.

DBOlderVersionThanService

The version of the AppLibrary Service schema on the database is incompatible with the version of the AppLibrary Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The AppLibrary Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the AppLibrary Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the AppLibrary and the database has been lost for an extended period of time, or has failed due to a configuration problem. The AppLibrary service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibDBConnection
- Test-AppLibDBConnection
- Get-AppLibDBConnection
- Get-AppLibDBSchema

New-AppLibAppVAppPublishedEvent

March 11, 2024

Records telemetry information

Syntax

```
New-AppLibAppVAppPublishedEvent
   -ApplicationUid <Int32>
   -ApplicationName <String>
   -RepositoryTypeValue <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Indicates that a packaged application is published to a delivery group.

Examples

EXAMPLE 1

Tracks that fact that BBC Ticker was published from a Microsoft App-V Server package repository

```
New-AppLibAppVAppPublishedEvent -ApplicationName "BBC Ticker" -
   RepositoryTypeValue 0 -ApplicationUid 23
```


EXAMPLE 2

Tracks that fact that BBC Ticker was published from a Citrix AppLibrary package repository

```
New-AppLibAppVAppPublishedEvent -ApplicationName "Notepad" -
    RepositoryTypeValue 1 -ApplicationUid 23
```

Parameters

-ApplicationUid

The broker application Uid

Type:	Int32
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationName

The application name

Type:	String
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RepositoryTypeValue

Indicates the source location of the application

Type:	Int32
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

Telemetry information is sent annonymously to CEIP (Customer Experiance Improvement Program) when the user has opted into the service.

Only the information specified in the parameters are sent to CEIP

This command is intended for internal use only.

RepositoryTypeValue(0) = Microsoft App-V Server (Dual Admin)

RepositoryTypeValue(1) = App-V Single Admin

RepositoryTypeValue(2) = MSIX

RepositoryTypeValue(3) = MSIX AppAttach

RepositoryTypeValue(4) = FlexApp

Related Links

- New-AppLibAppVPackageRepositoryEvent

New-AppLibAppVPackage

March 11, 2024

Adds a new package to the library.

Syntax

```
New-AppLibAppVPackage
   -LibraryUid <Int32>
   -Path <String>
   [-RetrieveIcon <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AppLibAppVPackage
   -Path <String>
   -AppManifestXml <String>
   -AppManifestIcons <System.Collections.Generic.Dictionary`2[System.
      String,System.Byte[]]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AppLibAppVPackage
   -Path <String>
   [-RetrieveIcon <Boolean>]
   -AppManifestXml <String>
   -AppManifestIcons <System.Collections.Generic.Dictionary`2[System.
      String,System.Byte[]]>
   -ManagementServer <String>
   -PublishingServer <String>
   -PackageProperties <System.Collections.Generic.Dictionary`2[System.
      String,System.Object]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The package and the applications that make it up are added to the library.

Examples

EXAMPLE 1

Adds the package at the specified network location to the library by interrogating the packge file and extrcting the neccesary informaion

```
New-AppLibAppVPackage -LibraryUid 1 -Path "\\NetworkStorage.enterprise.
   com\Managed App-V Packages\Package.appv"
```


EXAMPLE 2

Adds the package at the specified network location to the library by interpreting only the supplied AppManifest Xml when no physical access to the package is possible.

```
New-AppLibAppVPackage -Path "\\NetworkStorage.enterprise.com\Managed
    App-V Packages\Package.appv" -AppManifestXml $AppManifestxml -
    AppManifestIcons $DictinaryOfIcons
```

Parameters

-LibraryUid

The UID of the library to which the package is added.

Type:	Int32
Aliases:	Uid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Path

The full path to the package on the network share.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AppManifestXml

The Package's AppManifest Xml

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AppManifestIcons

A dictionary containing data for icons containied in te package keyed by their path in the AppManifest Xml

Type:	System.Collections.Generic.Dictionary‘2[System.String,Syste
Position	Named
Default value	None
Required	True
Accept pipeline input	False
Accept wildcard characters	False

-ManagementServer

Fully qualified domain name of the machine hosting Microsoft App-V Management Server

Type	String
Position	Named
Default value	None
Required	True
Accept pipeline input	True (ByPropertyName)
Accept wildcard characters	True

-PublishingServer

Url of App-V publishing server

Type	String
Position	Named
Default value	None
Required	True
Accept pipeline input	True (ByPropertyName)
Accept wildcard characters	True

-PackageProperties

Extracted AppV Package properties

Type	System.Collections.Generic.Dictionary‘2[System.String,Syste
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RetrieveIcon

A switch to indicate whether to return the package and applications’ icon data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.AppVPackage

An object representing the details of the the new package.

Notes

If supplied, the value of the LibraryUid parameter must be 1. No other values are supported.

When a package is added to the library and a record of that package already exists, the existing package is updated with the details of the new package. A package is considered as already existing when the following are true: 1) The PackageGuid and VersionGuid are the same as the existing package and the Path is the same or different. 2) The Path is the same as the existing package and the PackageGuid and/or VersionGuid are the same or different.

Related Links

- Remove-AppLibAppVPackage

New-AppLibAppVPackageRepositoryEvent

March 11, 2024

Records telemetry information

Syntax

```
New-AppLibAppVPackageRepositoryEvent
   [-RepositoryType <Int32>]
   [-PackageCount <Int32>]
   [-Location <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Indicates that an App-V package repository is added to Citrix Studio.

Examples

EXAMPLE 1

Records that a Microsoft App-V Server was used in Citrix Studio and that it contains 32 packages located at "`http://appvserver.acme.com:9000`"


```
New-AppLibAppVPackageRepositoryEvent -RepositoryType 1 -PackageCount 32
    -Location "http://appvserver.acme.com:9000"
```

EXAMPLE 2

Records that the internal Citrix AppLibrary was used in Citrix Studio and that it contains 32 packages.

```
New-AppLibAppVPackageRepositoryEvent -RepositoryType 2 -PackageCount 32
    -Location "AppLibrary"
```

Parameters

-RepositoryType

Indicates the source location of the application

Type:	Int32
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PackageCount

The number of packages in the repository

Type:	Int32
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Location

The location(URL) of the package repository

Type:	String
Position:	Named
Default value:	None - This is a mandatory parameter
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Guid

Unique identifier for this repository source

Notes

Telemetry information is sent annonymously to CEIP (Customer Experiance Improvement Program) when the user has opted into the service.

Only the information specified in the parameters are sent to CEIP

This command is intended for internal use only. The user is not able to add new package repositories to the system.

RepositoryTypeValue(0) = Microsoft App-V Server (Dual Admin)

RepositoryTypeValue(1) = App-V Single Admin

RepositoryTypeValue(1) = App-V Single Admin

RepositoryTypeValue(2) = MSIX

RepositoryTypeValue(3) = MSIX AppAttach

RepositoryTypeValue(4) = FlexApp

Related Links

- New-AppLibAppVAppPublishedEvent

New-AppLibIsolationGroup

March 11, 2024

Adds a new isolation group into the library

Syntax

```
New-AppLibIsolationGroup
   [-LibraryUid] <Int32>
   [-Name] <String>
   [-Description <String>]
   [-Version <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The isolation group will be added to the library

Examples

EXAMPLE 1

Adds the isolation group to the specified library

```
New-AppLibIsolationGroup -LibraryUid 1 -Name "MyIsolationGroup" -
   Description "Some description here" -Version "1.0.0.1"
```

Parameters

-LibraryUid

The unique identifier of the library to which the isolation group will be added

Type:	Int32
Aliases:	Uid
Accepted values:	1, 3, 4
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

The name of the isolation group.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

The description of the isolation group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Version

The version of the isolation group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.IsolationGroup

An object representing the details of the new isolation group.

Notes

If supplied, the value of the LibraryUid parameter must be 1. No other values are supported.

Related Links

- Remove-AppLibIsolationGroup
- Set-AppLibIsolationGroup
- Set-AppLibIsolationGroupPackage

New-AppLibPackageDiscovery

March 11, 2024

Starts a new package discovery session running on a VDA chosen from the specified desktop group

Syntax

```
New-AppLibPackageDiscovery
   -DiscoveryProfileUid <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AppLibPackageDiscovery
   -DesktopGroupUid <Int32>
   -Path <String>
   [-Recurse <Boolean>]
   [-DiscoverAppV <Boolean>]
   [-DiscoverMsix <Boolean>]
   [-DiscoverAppAttach <Boolean>]
   [-DiscoverFlexApp <Boolean>]
   [-CleanupAbsentPackages <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AppLibPackageDiscovery
   -DesktopGroupUid <Int32>
   -ManagementServer <String>
   -PublishingServer <String>
   [-Username <String>]
   [-Password <SecureString>]
   [-CleanupAbsentPackages <Boolean>]
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

A package discovery object has information about the state and progress of the discovery session

Examples

EXAMPLE 1

Initiates a new package discovery session that will discover packages in the specified folder and its subfolder tree

```
$BrokerDesktopGroup = Get-BrokerDesktopGroup -Name 'Package Discovery
    Group'
New-AppLibPackageDiscovery -DesktopGroupUid $BrokerDesktopGroup.Uid -
    Path "\\FileServer.company.com\AppVShare" -Recurse $true
```

EXAMPLE 2

Initiates a new package discovery session according to the options specified in the stored package discovery profile

```
$PackageDiscoveryProfile = Get-PackageDiscoveryProfile -Uid 1
New-AppLibPackageDiscovery -DiscoveryProfileUid
    $PackageDiscoveryProfile.Uid
```

Parameters

-DiscoveryProfileUid

The Uid of the package discovery profile to use when determining the parameters for the discovery session

Type:	Int32
Position:	Named
Default value:	0
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

The Uid of the desktop group from which the broker will select a VDA to run the discovery

Type:	Int32
Position:	Named
Default value:	0
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Path

The UNC path of the shared network folder (or file) location where the discovery will start

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ManagementServer

The URL of the App-V Management Server where the discovery will take place

Type:	String
Position:	Named
Default value:	Null

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishingServer

The URL of the App-V Publishing Server that will be used to publish the discovered packages

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Recurse

A value indicating whether the discovery will search through the child folder tree from the discovery root.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverAppV

A value indicating whether the discovery will include App-V packages

Type:	Boolean

Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverMsix

A value indicating whether the discovery will include Msix packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverAppAttach

A value indicating whether the discovery will include AppAttach packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverFlexApp

A value indicating whether the discovery will include FlexApp packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CleanupAbsentPackages

A value indicating whether the discovery should attempt to remove packages from the library that were not present in the discovery location

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Username

The username of the App-V server administrator

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password of the App-V server administrator

Type:	SecureString
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.PackageDiscovery

Informaton about the discovery session that was initiated

Notes

The discovery root path and a pair of Management and Publishing servers are mutually exclusive. I.e. a discovery session will run against either a network folder or an App-V Management and Publishing server

Related Links

- New-AppLibPackageDiscoveryProfile
- Get-AppLibPackageDiscoveryProfile
- Set-AppLibPackageDiscoveryProfile
- Remove-AppLibPackageDiscoveryProfile
- Get-AppLibPackageDiscovery

New-AppLibPackageDiscoveryProfile

March 11, 2024

Creates a new package discovery profile in the AppLibrary

Syntax

```
New-AppLibPackageDiscoveryProfile
   -Name <String>
   -DesktopGroupUid <Int32>
   -Path <String>
```



```
    [-Recurse <Boolean>]
    [-DiscoverAppV <Boolean>]
    [-DiscoverMsix <Boolean>]
    [-DiscoverAppAttach <Boolean>]
    [-DiscoverFlexApp <Boolean>]
    [-AutomateDiscovery <Boolean>]
    [-AutoDiscoveryPeriod <String>]
    [-AutoDiscoveryCadence <Int32>]
    [-CleanupAbsentPackages <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-AppLibPackageDiscoveryProfile
   -Name <String>
   -DesktopGroupUid <Int32>
   -ManagementServer <String>
   -PublishingServer <String>
   -Username <String>
   -Password <SecureString>
   [-AutomateDiscovery <Boolean>]
   [-AutoDiscoveryPeriod <String>]
   [-AutoDiscoveryCadence <Int32>]
   [-CleanupAbsentPackages <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The package discovery profile tells the AppLibrary service where to look for packages to autmatically inport to the AppLibrary, what types of packages should be included and how often the discovery should be done

Examples

EXAMPLE 1

Creates a new package discovery profile that will be automatically run once per day by the AppLibrary Service

```
New-AppLibPackageDiscoveryProfile -Name "App-V Package Share" -
    DesktopGroupUid $BrokerDesktopGroup.Uid -Path "\\FileServer.company.
    com\AppVShare" -Recurse $true -AutomateDiscovery $true -
    AutoDiscoveryPeriod "Day" AutoDiscoveryCadence "1"
```


EXAMPLE 2

Creates a new package discovery profile that will target an App-V Management and Publishing server, but will not run automatically

```
1 $SecurePassword = ConvertTo-SecureString -String "password" -
     AsPlainText -Force
2 New-AppLibPackageDiscoveryProfile -Name "App-V Server" -DesktopGroupUid
      $BrokerDesktopGroup.Uid -ManagementServer "http://AppVServer.
     company.com" -PublishingServer "http://AppVServer.company.com:8001"
     -Username "AppVServer\Username" -Password $SecurePassword -
     AutomateDiscovery $false
```

Parameters

-Name

The name of the package discovery profile

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

The Uid of the desktop group from which the broker will select a VDA to run the discovery

Type:	Int32
Position:	Named
Default value:	0
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Path

The UNC path of the shared network folder (or file) location where the discovery will start

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ManagementServer

The URL of the App-V Management Server where the discovery will take place

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishingServer

The URL of the App-V Publishing Server that will be used to publish the discovered packages

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Username

The username of the App-V server administrator

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password of the App-V server administrator

Type:	SecureString
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Recurse

A value indicating whether the discovery will search through the child folder tree from the discovery root

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverAppV

A value indicating whether the discovery will include App-V packages

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverMsix

A value indicating whether the discovery will include Msix packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverAppAttach

A value indicating whether the discovery will include AppAttach packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DiscoverFlexApp

A value indicating whether the discovery will include FlexApp packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomateDiscovery

A value indicating whether the discovery will run automatically according to the configured discovery cadence

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoDiscoveryPeriod

The period of time to measure the discovery cadence in E.g. Hours, Days or Weeks.

Type:	String
Position:	Named
Default value:	Day
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoDiscoveryCadence

The minimum number of periods that should elaps between discovery sessions

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CleanupAbsentPackages

A value indicating whether the discovery should attempt to remove packages from the library that were not present in the discovery location

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.PackageDiscoveryProfile

An object representing the configured discovery profile

Notes

The discovery root path and pair of Management and Publishing servers are mutually exclusive. I.e. a discovery session will run against either a network folder or an App-V Management and Publishing server

Setting a discovery cadence of 0 (on any period) is equivilent to setting AutomateDiscovery to False

Related Links

- Get-AppLibPackageDiscoveryProfile
- Set-AppLibPackageDiscoveryProfile
- Remove-AppLibPackageDiscoveryProfile
- New-AppLibPackageDiscovery
- Get-AppLibPackageDiscovery

Remove-AppLibAppVApplicationMetadata

March 11, 2024

Removes metadata from the given AppVApplication.

Syntax

```
Remove-AppLibAppVApplicationMetadata
      [-AppVApplicationUid] <Int32>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVApplicationMetadata
      [-AppVApplicationUid] <Int32>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVApplicationMetadata
      [-AppVApplicationName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVApplicationMetadata
      [-AppVApplicationName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-AppLibAppVApplicationMetadata
      [-InputObject] <AppVApplication[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVApplicationMetadata
      [-InputObject] <AppVApplication[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVApplicationMetadata
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given AppVApplication.

Examples

EXAMPLE 1

Removes the metadata item named ‘Custom Data’from the specified AppVApplication object.

```
Remove-AppLibAppVApplicationMetadata -AppVApplicationUid 1 -Name "
    Custom Data"
```

EXAMPLE 2

Remove all metadata from all AppVApplication objects.

```
Get-AppLibAppVApplication | % {
  Remove-AppLibAppVApplicationMetadata -Map $_.MetadataMap  }
```


Parameters

-AppVApplicationUid

Id of the AppVApplication

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-AppVApplicationName

Name of the AppVApplication

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects from which the metadata is to be removed.

Type:	AppVApplication[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibAppVApplicationMetadata

Remove-AppLibAppVPackage

March 11, 2024

Removes an App-V package from the library that is holding it

Syntax

```
Remove-AppLibAppVPackage
      [-Uid] <Int32>
      [-ServerUid <Int32>]
      [-Purge <Boolean>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove an App-V package from the application library.

Removing a package from the library optionally only marks it as not existing. In this case, re-adding the package later will restore the reference.

Examples

EXAMPLE 1

Permanently removes the specified package from the library including its associated applications and metadata.

```
Remove-AppLibAppVPackage -Uid 5
```

EXAMPLE 2

Permanently removes the specified package from the library including its associated applications and metadata.

```
Remove-AppLibAppVPackage -Uid 5 -Purge $true
```


EXAMPLE 3

Removes the specified package from the library that holds it by marking it as non-existing, but leaving associated records and metadata in place.

```
1  Remove-AppLibAppVPackage -Uid 5 -Purge $false
```

Parameters

-Uid

The App Library's internal unique identifier of the App-V package.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ServerUid

Unique Identifier for AppVServer.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Purge

Indicates whether the package should be removed from the database. The default is true. Setting this parameter to false will leave the package and all of its associated applications and metadata in place but in a dormant state. This data will then be reactivated if the same package is ever re-added.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can result.

Error Codes

UnknownObject

No package was found for the given uid. DatabaseError

An error occurred in the service while attempting a database operation. DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured. DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons. PermissionDenied

You do not have permission to execute this command. AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service. CommunicationError

There was a problem communicating with the remote service. ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

Remove-AppLibAppVPackageMetadata

March 11, 2024

Removes metadata from the given AppVPackage.

Syntax

```
Remove-AppLibAppVPackageMetadata
      [-AppVPackageUid] <Int32>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
      [-AppVPackageUid] <Int32>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
      [-AppVPackageName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
      [-AppVPackageName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
      [-InputObject] <AppVPackage[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
      [-InputObject] <AppVPackage[]>
```



```
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-AppLibAppVPackageMetadata
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given AppVPackage.

Examples

EXAMPLE 1

Removes the metadata item named ‘Custom Data’from the specified AppVPackage object.

```
Remove-AppLibAppVPackageMetadata -AppVPAckageUid 1 -Name "Custom Data"
```

EXAMPLE 2

Remove all metadata from all AppVPackage objects.

```
Get-AppLibAppVPackage | % {
  Remove-AppLibAppVPackageMetadata -Map $_.MetadataMap  }
```

Parameters

-AppVPackageUid

Id of the AppVPackage

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-AppVPackageName

Name of the AppVPackage

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects from which the metadata is to be removed.

Type:	AppVPackage[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibAppVPackageMetadata

Remove-AppLibAppVServer

March 11, 2024

Removes an App-V server information

Syntax

```
Remove-AppLibAppVServer
       [-Uid] <Int32>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Provides the ability to remove an App-V server, associated packages and applications.

Examples

EXAMPLE 1

Removes the specified AppVServer, associated packages and applications from DDC

```
1  Remove-AppLibAppVServer -Uid 5
```

Parameters

-Uid

Id of AppVServer.

Type:	Int32
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-AppLibIsolationGroup

March 11, 2024

Removes an isolation group from the library

Syntax

```
Remove-AppLibIsolationGroup
      [-IsolationGroupUid] <Int32>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The isolation group will be removed from the library that is holding it

Examples

EXAMPLE 1

Removes the specified isolation group from the application library that holds it.

```
Remove-AppLibIsolationGroup -IsolationGroupUid 5
```

Parameters

-IsolationGroupUid

The isolation group's internal unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-AppLibIsolationGroupMetadata

March 11, 2024

Removes metadata from the given IsolationGroup.

Syntax

```
Remove-AppLibIsolationGroupMetadata
      [-IsolationGroupUid] <Int32>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
      [-IsolationGroupUid] <Int32>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
      [-IsolationGroupName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
      [-IsolationGroupName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
      [-InputObject] <IsolationGroup[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
      [-InputObject] <IsolationGroup[]>
```



```
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-AppLibIsolationGroupMetadata
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given IsolationGroup.

Examples

EXAMPLE 1

Removes the metadata item named 'Custom Data'from the specified IsolationGroup object.

```
Remove-AppLibIsolationGroupMetadata -IsolationGroupUid 1 -Name "Custom
    Data"
```

EXAMPLE 2

Remove all metadata from all IsolationGroup objects.

```
Get-AppLibIsolationGroup | % {
  Remove-AppLibIsolationGroupMetadata -Map $_.MetadataMap  }
```

Parameters

-IsolationGroupUid

Id of the IsolationGroup

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IsolationGroupName

Name of the IsolationGroup

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects from which the metadata is to be removed.

Type:	IsolationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibIsolationGroupMetadata

Remove-AppLibIsolationGroupPackage

March 11, 2024

Removes an AppV package from an isolation group

Syntax

```
Remove-AppLibIsolationGroupPackage
      [-IsolationGroupUid] <Int32>
      -AppVPackageUid <Int32>
      [-LoggingId <Guid>]
```



```
5         [<CitrixCommonParameters>]
6         [<CommonParameters>]
```

Description

Removes the App-V package with the unique identifier in the specified isolation group

Examples

EXAMPLE 1

Removes the App-V package with the unique identifier in the specified isolation group.

```
1  Remove-AppLibIsolationGroupPackage -IsolationGroupUid 4 -AppVPackageUid
       15
```

Parameters

-IsolationGroupUid

The isolation group's internal unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-AppVPackageUid

The unique identifier of the package within the isolation group.

Type:	Int32
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-AppLibLibraryMetadata

March 11, 2024

Removes metadata from the given Library.

Syntax

```
Remove-AppLibLibraryMetadata
      [-LibraryUid] <Int32>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
      [-LibraryUid] <Int32>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
      [-LibraryName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
      [-LibraryName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
```



```
        [-InputObject] <Library[]>
        -Name <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
        [-InputObject] <Library[]>
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-AppLibLibraryMetadata
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Library.

Examples

EXAMPLE 1

Removes the metadata item named ‘Custom Data’from the specified Library object.

```
Remove-AppLibLibraryMetadata -LibraryUid 1 -Name "Custom Data"
```

EXAMPLE 2

Remove all metadata from all Library objects.

```
Get-AppLibLibrary | % {
  Remove-AppLibLibraryMetadata -Map $_.MetadataMap  }
```

Parameters

-LibraryUid

Id of the Library

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LibraryName

Name of the Library

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects from which the metadata is to be removed.

Type:	Library[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibLibraryMetadata

Remove-AppLibPackageDiscoveryProfile

March 11, 2024

Removes a package discovery profile from the AppLibrary service

Syntax

```
Remove-AppLibPackageDiscoveryProfile
      [-Uid] <Int32>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The package discovery profile will be removed from the AppLibrary service

Examples

EXAMPLE 1

Removes the specified package discovery profile from the AppLibrary service

```
Remove-AppLibPackageDiscoveryProfile -Uid 5
```

Parameters

-Uid

The package discovery profile's internal unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-AppLibPackageDiscoveryProfile

- Get-AppLibPackageDiscoveryProfile
- Set-AppLibPackageDiscoveryProfile
- New-AppLibPackageDiscovery
- Get-AppLibPackageDiscovery

Remove-AppLibServiceConfigurationData

March 11, 2024

Removes configuration data from the service.

Syntax

```
Remove-AppLibServiceConfigurationData
       [[-Name] <String>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove data from the AppLibrary Service configuration data.

Examples

EXAMPLE 1

Removes the configuration data item with name “MyName”.

```
Remove-AppLibServiceConfigurationData -Name "MyName"
```

Parameters

-Name

The name of the configuration data item to remove.

Type: String

Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibServiceConfigurationData
- Get-AppLibServiceConfigurationData

Remove-AppLibServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-AppLibServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AppLibServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
```



```
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-AppLibService | % {
  Remove-AppLibServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Set-AppLibServiceMetadata

Reset-AppLibEnabledFeatureList

March 11, 2024

Refreshes the AppLibrary service's list of enabled features.

Syntax

```
Reset-AppLibEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix AppLibrary Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. However, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected AppLibrary service instance's list of enabled features.

```
Reset-AppLibEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-AppLibServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the AppLibrary Service.

Syntax

```
Reset-AppLibServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload AppLibrary Service access permissions and configuration service locations. The Reset-AppLibServiceGroupMembership command must be run on at least one instance of the service type (AppLib) after installation and registration with the configuration service. Without

this operation, the AppLibrary services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AppLibServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    AppLibServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-AppLibServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.AppLibrary.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AppLibServiceGroupMembership command.

Outputs

Citrix.AppLibrary.Sdk.ServiceInstance

Reset-AppLibServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the AppLibrary Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibServiceInstance
- Get-AppLibServiceStatus

Set-AppLibAppVApplicationMetadata

March 11, 2024

Adds or updates metadata on the given AppVApplication.

Syntax

```
Set-AppLibAppVApplicationMetadata
   [-AppVApplicationUid] <Int32>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVApplicationMetadata
   [-AppVApplicationUid] <Int32>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVApplicationMetadata
   [-AppVApplicationName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVApplicationMetadata
   [-AppVApplicationName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVApplicationMetadata
```



```
    [-InputObject] <AppVApplication[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AppLibAppVApplicationMetadata
    [-InputObject] <AppVApplication[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given AppVApplication objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the AppVApplication with the Uid 5.

```
Set-AppLibAppVApplicationMetadata -AppVApplicationUid 5 -Name property
    -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-AppVApplicationUid

Id of the AppVApplication

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	False

-AppVApplicationName

Name of the AppVApplication

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	AppVApplication[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the AppVApplication specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AppLibAppVApplicationMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibAppVApplicationMetadata

Set-AppLibAppVPackageMetadata

March 11, 2024

Adds or updates metadata on the given AppVPackage.

Syntax

```
Set-AppLibAppVPackageMetadata
   [-AppVPackageUid] <Int32>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVPackageMetadata
   [-AppVPackageUid] <Int32>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVPackageMetadata
   [-AppVPackageName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVPackageMetadata
   [-AppVPackageName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVPackageMetadata
   [-InputObject] <AppVPackage[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibAppVPackageMetadata
   [-InputObject] <AppVPackage[]>
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given AppVPackage objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the AppVPackage with the Uid 5.

```
Set-AppLibAppVPackageMetadata -AppVPackageUid 5 -Name property -Value
    value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-AppVPackageUid

Id of the AppVPackage

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-AppVPackageName

Name of the AppVPackage

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	AppVPackage[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the AppV-Package specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AppLibAppVPackageMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibAppVPackageMetadata

Set-AppLibDBConnection

March 11, 2024

Configures a database connection for the AppLibrary Service.

Syntax

```
Set-AppLibDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix AppLibrary Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-AppLibDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-AppLibDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the AppLibrary Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-AppLibDBConnection cmdlet returns an object describing the status of the AppLibrary Service together with extra diagnostics information. Possible values are:

- OK:

The AppLibrary Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the AppLibrary Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the AppLibrary Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the AppLibrary Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the AppLibrary Service currently in use is newer than, and incompatible with, the version of the AppLibrary Service schema on the database. Upgrade the AppLibrary Service to a more recent version.

- DBOlderVersionThanService:

The version of the AppLibrary Service schema on the database is newer than, and incompatible with, the version of the AppLibrary Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the AppLibrary Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the AppLibrary Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the AppLibrary Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibServiceStatus
- Get-AppLibDBConnection
- Test-AppLibDBConnection

Set-AppLibDBCredentials

March 11, 2024

Configures the database server SQL credentials for the AppLibrary Service.

Syntax

```
Set-AppLibDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Specifies SQL credentials to be used by the currently selected Citrix AppLibrary Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional AppLibrary Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-AppLibDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-AppLibDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-AppLibDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.


```
Set-AppLibDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AppLibDBSchema
- Set-AppLibDBConnection
- Get-Credential

Set-AppLibIsolationGroup

March 11, 2024

Updates an isolation group in the library

Syntax

```
Set-AppLibIsolationGroup
   [-IsolationGroupUid] <Int32>
   [-Name <String>]
   [-Description <String>]
   [-Version <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The isolation group will be updated with the new properties

Examples

EXAMPLE 1

Updates only the name for the specified isolation group.

```
Set-AppLibIsolationGroup -IsolationGroupUid 58 -Name "MyIsolationGroup"
```


EXAMPLE 2

Updates only the description for the specified isolation group.

```
Set-AppLibIsolationGroup -IsolationGroupUid 58 -Description "Some
    description here"
```

EXAMPLE 3

Updates only the version for the specified isolation group.

```
Set-AppLibIsolationGroup -IsolationGroupUid 58 -Version "1.0.0.1"
```

EXAMPLE 4

Updates the name, description and version for the specified isolation group.

```
Set-AppLibIsolationGroup -IsolationGroupUid 58 -Name "MyIsolationGroup"
     -Description "Some description here" -Version "1.0.0.1"
```

Parameters

-IsolationGroupUid

The AppLibrary's internal unique identifier of the Isolation Group.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

The name of the isolation group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

The description of the isolation group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Version

The version of the isolation group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

At least one of the name, description and version parameters must be supplied.

Related Links

- New-AppLibIsolationGroup
- Remove-AppLibIsolationGroup
- Set-AppLibIsolationGroupPackage

Set-AppLibIsolationGroupMetadata

March 11, 2024

Adds or updates metadata on the given IsolationGroup.

Syntax

```
Set-AppLibIsolationGroupMetadata
   [-IsolationGroupUid] <Int32>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibIsolationGroupMetadata
   [-IsolationGroupUid] <Int32>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibIsolationGroupMetadata
   [-IsolationGroupName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibIsolationGroupMetadata
   [-IsolationGroupName] <String>
   -Map <PSObject>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AppLibIsolationGroupMetadata
    [-InputObject] <IsolationGroup[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AppLibIsolationGroupMetadata
    [-InputObject] <IsolationGroup[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given IsolationGroup objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the IsolationGroup with the Uid of 5.

```
Set-AppLibIsolationGroupMetadata -IsolationGroupUid 5 -Name property -
    Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-IsolationGroupUid

Id of the IsolationGroup

Type: Int32

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-IsolationGroupName

Name of the IsolationGroup

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	IsolationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the IsolationGroup specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AppLibIsolationGroupMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibIsolationGroupMetadata

Set-AppLibIsolationGroupPackage

March 11, 2024

Updates an App-V package's settings in an Isolation Group.

Syntax

```
Set-AppLibIsolationGroupPackage
   [-IsolationGroupUid] <Int32>
   -AppVPackageUid <Int32>
   -OrderNumber <Int32>
   -ExplicitInclusion <Boolean>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Updates an App-V package's settings in the specified Isolation Group.

Examples

EXAMPLE 1

Updates the App-V package with the unique identifier in the specified isolation group.

```
Set-AppLibIsolationGroupPackage -IsolationGroupUid 4 -AppVPackageUid 15
    -OrderNumber 1 -ExplicitInclusion $true
```


Parameters

-IsolationGroupUid

The AppLibrary's internal unique identifier of the Isolation Group to contain the package.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppVPackageUid

The AppLibrary's internal unique identifier of the App-V package.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OrderNumber

The order number within the Isolation Group.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-ExplicitInclusion

Indicates whether the package is explicitly included in the delivery group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Remove-AppLibIsolationGroupPackage

Set-AppLibLibraryMetadata

March 11, 2024

Adds or updates metadata on the given Library.

Syntax

```
Set-AppLibLibraryMetadata
   [-LibraryUid] <Int32>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-AppLibLibraryMetadata
   [-LibraryUid] <Int32>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibLibraryMetadata
   [-LibraryName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibLibraryMetadata
   [-LibraryName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibLibraryMetadata
   [-InputObject] <Library[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibLibraryMetadata
   [-InputObject] <Library[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Library objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Library with the Uid 1.


```
Set-AppLibLibraryMetadata -LibraryUid 1 -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-LibraryUid

Id of the Library

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-LibraryName

Name of the Library

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Library[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Library specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AppLibLibraryMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibLibraryMetadata

Set-AppLibPackageDiscoveryProfile

March 11, 2024

Updates a package discovery profile defined in the AppLibrary with the specified new details

Syntax

```
Set-AppLibPackageDiscoveryProfile
   -Uid <Int32>
   [-Name <String>]
   [-DesktopGroupUid <Int32>]
```



```
    -Path <String>
    [-Recurse <Boolean>]
    [-DiscoverAppV <Boolean>]
    [-DiscoverMsix <Boolean>]
    [-DiscoverAppAttach <Boolean>]
    [-DiscoverFlexApp <Boolean>]
    [-AutomateDiscovery <Boolean>]
    [-AutoDiscoveryPeriod <String>]
    [-AutoDiscoveryCadence <Int32>]
    [-CleanupAbsentPackages <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AppLibPackageDiscoveryProfile
   -Uid <Int32>
    [-Name <String>]
    [-DesktopGroupUid <Int32>]
    [-Recurse <Boolean>]
    [-ManagementServer <String>]
    [-PublishingServer <String>]
    [-Username <String>]
    [-Password <SecureString>]
    [-AutomateDiscovery <Boolean>]
    [-AutoDiscoveryPeriod <String>]
    [-AutoDiscoveryCadence <Int32>]
    [-CleanupAbsentPackages <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

A package discovery can be used by the AppLibrary Service to automatically (or periodically) search for application packages to import into the AppLibrary

Examples

EXAMPLE 1

Updates the package discovery profile with a Uid of 1 with the supplied values

```
Set-PackageDiscoveryProfile -Uid 1 Name "App-V Package Shere" -
    DesktopGroupUid $BrokerDesktopGroup.Uid -Path "\\FileServer.company.
    com\AppVShare" -Recurse $true -AutomateDiscovery $true -
    AutoDiscoveryPeriod "Day" AutoDiscoveryCadence "1"
```


EXAMPLE 2

Updates the package discovery profile by changing the properties on the object and passing it to the pipeline

```
$PackageDiscoveryProfile = Get-PackageDiscoveryProfile -Uid 1
$PackageDiscoveryProfile.AutoDiscoveryPeriod = "Hour"
$PackageDiscoveryProfile.AutoDiscoveryCadence "6"
$PackageDiscoveryProfile | Set-PackageDiscoveryProfile
```

Parameters

-Uid

The Uid of the package discovery profile that will be updated with the new details

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Path

The UNC path of the shared network folder (or file) location where the discovery will start

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Name

The name of the package discovery profile

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

The Uid of the desktop group from which the broker will select a VDA to run the discovery

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DiscoverAppV

A value indicating whether the discovery will include App-V packages

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DiscoverMsix

A value indicating whether the discovery will include Msix packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DiscoverAppAttach

A value indicating whether the discovery will include AppAttach packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DiscoverFlexApp

A value indicating whether the discovery will include FlexApp packages

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutomateDiscovery

A value indicating whether the discovery will run automatically according to the configured discovery cadence

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoDiscoveryPeriod

The period of time to measure the discovery cadence in E.g. Hours, Days or Weeks

Type:	String
Position:	Named
Default value:	Day
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AutoDiscoveryCadence

The number of periods that should elaps between discovery sessions

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-CleanupAbsentPackages

A value indicating whether the discovery should attempt to remove packages from the library that were not present in the discovery location

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ManagementServer

The URL of the App-V Management Server where the discovery will take place

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PublishingServer

The URL of the App-V Publishing Server that will be used to publish the discovered packages

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Username

The username of the App-V server administrator

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password of the App-V server administrator

Type:	SecureString
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Recurse

A value indicating whether the discovery will search through the child folder tree from the discovery root

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-AppLibPackageDiscoveryProfile
- Get-AppLibPackageDiscoveryProfile
- Remove-AppLibPackageDiscoveryProfile
- New-AppLibPackageDiscovery
- Get-AppLibPackageDiscovery

Set-AppLibServiceConfigurationData

March 11, 2024

Sets the value for the given key in the service configuration data.

Syntax

```
Set-AppLibServiceConfigurationData
   [-Name] <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored for the AppLibrary Service.

Examples

EXAMPLE 1

Set data with a name of 'customProperty1'and value of 'value2'to the service configuration.

```
Set-AppLibServiceConfigurationData -Name "customProperty1" -Value "
   value2"

Name                           Value
--------                       -----
customProperty1                value2
```


Parameters

-Name

Specifies the key name of the metadata to be added. The key must be unique.

The Name cannot contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the name. If the name already exists, its value is updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.AppLibrary.Sdk.ServiceConfigurationData

Set-AppLibServiceConfigurationData returns an object containing the new definition of the configuration.

Name <string>

Specifies the name for the item of data.

Value <string>

Specifies the value of the data.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibServiceConfigurationData
- Get-AppLibServiceConfigurationData

Set-AppLibServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-AppLibServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AppLibServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-AppLibServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1"= "val1"; "name2"= "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

Type:	PSObject
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AppLibServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Remove-AppLibServiceMetadata

Test-AppLibDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix AppLibrary Service.

Syntax

```
Test-AppLibDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AppLibDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AppLibDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix AppLibrary Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a AppLibrary SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1  Test-AppLibDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
       Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix AppLibrary Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-AppLibDBConnection cmdlet returns an object describing the status of the selected AppLibrary Service instance that would result if the connection string were used with the Set-AppLibDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-AppLibDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the AppLibrary Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the AppLibrary Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The AppLibrary Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The AppLibrary Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the AppLibrary Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the AppLibrary Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AppLibAppLibrarySnapin
- Get-AppLibServiceStatus
- Get-AppLibDBConnection
- Set-AppLibDBConnection

Close-CtxAppVServerSession

March 11, 2024

Syntax

```
Close-CtxAppVServerSession
     [<CommonParameters>]
```

Description

Description

Examples

Parameters

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Related Links

Close-CtxAppVServerSession

March 11, 2024

Syntax

```
Close-CtxAppVServerSession
     [<CommonParameters>]
```

Description

Description

Examples

Parameters

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Related Links

ConvertTo-CtxAppVLauncherArg

March 11, 2024

Returns a string containing information to send to the App-V Launcher. You can plug this string directly into the Virtual Delivery Agent (VDA) to launch App-V applications.

Syntax

```
ConvertTo-CtxAppVLauncherArg
          [-AppVPublishingServer] <String>
          [-PackageId] <String>
          [-AppId] <String>
          -SeqLoc <String>
          [-CmdLineArg <String>]
          -TargetInPackage <Boolean>
          -ApplicationName <String>
          [<CommonParameters>]
```

```
ConvertTo-CtxAppVLauncherArg
          [-LauncherPath]
          [<CommonParameters>]
```

Description

Returns a string containing information to send to the App-V Launcher. You can plug this string directly into the Virtual Delivery Agent (VDA) to launch App-V applications.

Examples

EXAMPLE 1

Converts the arguments provided into a cmdlet to launch Beyond Compare from the appv-pubsrv Publishing Server on the VDA.


```
ConvertTo-CtxAppVLauncherArg -AppVPublishingServer "http://appv-pubsrv.
    mydomain.com:8082" -PackageId "1bcb6993-10b1-4659-b9d4-
    e809e10cecdf_5" -AppId "[{
 ProgramFilesX86 }
 ]\Beyond Compare 3\BCompare.exe" -SeqLoc "\\10.105.32.199\share\App-V\
     Firefox.appv" -TargetInPackage $true
```

EXAMPLE 2

Gets the Citrix App-V Launcher path.

```
ConvertTo-CtxAppVLauncherArg -LauncherPath
```

Parameters

-AppVPublishingServer

The Url including the port number, of the publishing Server which is serving the application.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-AppId

The AppId of the specific application for which CtxAppVLauncher argugents is required.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PackageId

The PackageId of the package to which the application belongs.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SeqLoc

The sequence location of the package.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetInPackage

Pass this as $true if TargetInPackage field is “true”in the Manifest file for the particular App Id.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LauncherPath

Gets the Citrix Launcher path. The Citrix Launcher is a component installed on the VDA.

Type:	SwitchParameter
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-ApplicationName

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-CmdLineArg

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Get-CtxAppVApplication

March 11, 2024

Enumerates all published App-V applications for a given Management Server.

Syntax

```
Get-CtxAppVApplication
    [-AppVManagementServer] <String>
    [<CommonParameters>]
```

Description

Queries a given Management Server and fetches all published applications for that server. Applications that are published but have no user entitlements are not displayed by this cmdlet.

Examples

EXAMPLE 1

Displays all applications available from the Management Server “appv-mansrv”.

```
Get-CtxAppVApplication -AppVManagementServer "appv-mansrv"
```

Parameters

-AppVManagementServer

Machine name of the App-V Management Server.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can’t pipe objects to this cmdlet.

Outputs

Citrix.VirtApp.Studio.PowerShellManager.AppVServerApplication

Related Links

Get-CtxAppVApplicationInfo

March 11, 2024

Enumerates information for a given application in a given package for a given Management Server.

Syntax

```
Get-CtxAppVApplicationInfo
   -AppVManagementServer <String>
    [[-Property] <String[]>]
    [-AppId] <String>
    [-PackageId] <String>
    [<CommonParameters>]
```

```
Get-CtxAppVApplicationInfo
   -AppVManagementServer <String>
    [[-Property] <String[]>]
    [-InputObject] <AppVServerApplication[]>
    [<CommonParameters>]
```

Description

Queries a given Management Server and fetches the requested information for a given application.

Examples

EXAMPLE 1

Fetches all application information for the ‘Beyond Compare 3’application from the appv-mansrv Management Server.

```
Get-CtxAppVApplicationInfo -AppVManagementServer "appv-mansrv" -AppId "
   [{
 ProgramFilesX86 }
 ]\Beyond Compare 3\BCompare.exe"  -PackageId "1bcb6993-10b1-4659-b9d4-
    e809e10cecdf_5"
```


EXAMPLE 2

Fetches only information about FTA and user entitlements for the Beyond Compare 3 application from the appv-mansrv Management Server.

```
Get-CtxAppVApplicationInfo -AppVManagementServer "appv-mansrv" -AppId "
    [{
 ProgramFilesX86 }
 ]\Beyond Compare 3\BCompare.exe"  -PackageId "1bcb6993-10b1-4659-b9d4-
    e809e10cecdf_5" -Property @('FTA','USERS')
```

EXAMPLE 3

Pipelines the output of Get-CtxAppVApplication to Get-CtxAppVApplicationInfo to get the application properties of all the published applications on the appv-mansrv Management Server.

```
Get-CtxAppVApplication -AppVManagementServer "appv-mansrv" | Get-
    CtxAppVApplicationInfo -AppVManagementServer "appv-mansrv"
```

Parameters

-AppId

The AppId of the specific application for which information is required.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PackageId

The PackageId of the package to which the application belongs.

Type:	String
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

AppVServerApplication objects for which information is required.

Type:	AppVServerApplication[]
Position:	4
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Property

An array of the names of the property values to be pupulated in the returned data.

Type:	String[]
Position:	4
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppVManagementServer

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.VirtApp.Studio.PowerShellManager.AppVAppData

Related Links

Get-CtxAppVPackagesEnabledCount

March 11, 2024

Syntax

```
Get-CtxAppVPackagesEnabledCount
   -AppVManagementServer <String>
   [<CommonParameters>]
```

Description

Description

Examples

Parameters

-AppVManagementServer

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

Outputs

Int32

Related Links

Get-CtxAppVServer

March 11, 2024

Returns URLs for App-V Publishing and Management Servers contained in a Citrix App-V policy. Returned values are in string format.

Syntax

```
Get-CtxAppVServer
   -ByteArray <Byte[]>
   [-ConsumedByStudio <Boolean>]
   [<CommonParameters>]
```

Description

Returns URLs for App-V Publishing and Management Servers contained in a Citrix App-V policy. Returned values are in string format.

Examples

EXAMPLE 1

Returns Publishing Server URL , Management Server URL configured in given policy

```
System.Management.Automation.PSObject[]
```

Parameters

-ByteArray

Specifies the Citrix App-V policy created using the New-CtxAppVServer cmdlet.

Type:	Byte[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-ConsumedByStudio

If set to “true”, outputs URLs for the App-V Publishing and Management Servers. If set to “false”, outputs both the URL and settings for the App-V Publishing Server.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

Related Links

Get-CtxAppVServerSetting

March 11, 2024

Returns settings for the specified App-V Publishing Server.

Syntax

```
Get-CtxAppVServerSetting
   -AppVPublishingServer <String>
   [<CommonParameters>]
```


Description

Returns settings for the specified App-V Publishing Server. For more information about these settings, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Examples

EXAMPLE 1

This example returns settings associated with `http://appv-pubsrv.mydomain.com:8082` in the following format:

GlobalRefreshEnabled: false

GlobalRefreshOnLogon: false

GlobalRefreshInterval: 0

GlobalRefreshIntervalUnit: Day

UserRefreshEnabled: true

UserRefreshOnLogon: false

UserRefreshInterval: 0

UserRefreshIntervalUnit: Hour

```
1  Get-CtxAppVServerSetting -AppVPublishingServer http://appv-pubsrv.
       mydomain.com:8082
```

Parameters

-AppVPublishingServer

The full URL (including port number) of the Publishing Server for which settings are to be returned.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	True

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.VirtApp.Studio.Commands.AppVServerSetting

Publishing Server settings. These settings are:

GlobalRefreshEnabled

GlobalRefreshOnLogon

GlobalRefreshInterval

GlobalRefreshIntervalUnit

UserRefreshEnabled

UserRefreshOnLogon

UserRefreshInterval

UserRefreshIntervalUnit

Related Links

New-CtxAppVServer

March 11, 2024

Creates a new Citrix App-V policy containing the specified App-V Management and Publishing Server URLs.

Syntax

```
New-CtxAppVServer
   -PublishingServer <String>
   -ManagementServer <String>
   [-UserRefreshEnabled <Boolean>]
   [-UserRefreshOnLogOn <Boolean>]
   [-UserRefreshInterval <Int32>]
   [-UserRefreshIntervalUnit <RefreshInterval>]
   [-GlobalRefreshEnabled <Boolean>]
   [-GlobalRefreshOnLogOn <Boolean>]
   [-GlobalRefreshInterval <Int32>]
   [-GlobalRefreshIntervalUnit <RefreshInterval>]
   [<CommonParameters>]
```

Description

Creates a new Citrix App-V policy containing the specified App-V Management and Publishing Server URLs. Additionally, accepts Publishing Server settings that control how and when automatic refresh occurs on the VDA.

Examples

EXAMPLE 1

Creates a new Citrix Policy for Management Server appv-mansrv & Publishing Server: appv-pubsrv on port 8082.

Default Publishing Server setting is used for `http://appv-pubsrv.mydomain.com:8082`

Default values for Publishing Server settings are:

GlobalRefreshEnabled = false

GlobalFreshOnLogon = false

GlobalIntervalRefreshInterval = 0

GlobalRefreshIntervalUnit = Day

UserRefreshEnabled = true

UserRefreshOnLogon = true

UserIntervalRefreshInterval = 0

GlobalRefreshIntervalUnit = Day

```
New-CtxAppVServer -ManagementServer http://appv-mansrv.mydomain.com -
    PublishingServer http://appv-pubsrv.mydomain.com:8082
```

EXAMPLE 2

Creates a new Citrix Policy for Management Server AppV-Mgmt-Server:8080 & Publishing Server: AppV-Mgmt-Server:8082. User specified Publishing Server settings are used for AppV-Mgmt-Server:8082

Following values are used to configure Publishing Server appv-pubsrv

GlobalRefreshEnabled = True

GlobalRefreshOnLogon = True

GlobalIntervalRefreshInterval = 2

GlobalRefreshIntervalUnit = Hour

UserRefreshEnabled = true

UserRefreshOnLogon = true

UserIntervalRefreshInterval = 3

GlobalRefreshIntervalUnit = Hour

```
New-CtxAppVServer -ManagementServer http://appv-mansrv.mydomain.com -
    PublishingServer http://appv-pubsrv.mydomaain.com:8082 -
    GlobalRefreshEnabled $true -GlobalRefreshOnLogon $true -
    GlobalRefreshInterval 2 -GlobalRefreshIntervalUnit Hour -
    UserRefreshEnabled $true -UserRefreshOnLogon $true -
    UserRefreshInterval 3 -UserRefreshIntervalUnit Hour
```

Parameters

-ManagementServer

The URL of the Management Server to add to the Citrix policy.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	True

-PublishingServer

The URL (including the port number) of the Publishing Server to add to the Citrix policy.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshEnabled

Enables a refresh of packages published to user groups either at user logon or at a specified interval. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshIntervalUnit

Specifies the unit for the UserRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	RefreshInterval
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshEnabled

Enables a refresh of packages published to machine groups either at user logon or at a specified interval. For more information, see the App-V 5.0 documentation at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to machine groups. This can be either days or hours, as specified by the GlobalRefreshIntervalUnit setting. Please refer to App-V 5.0 documentation for details. http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshIntervalUnit

Specifies the unit for the GlobalRefreshInterval setting. This can be set to either Hours (0) or Days (1). Please refer to App-V 5.0 documentation for details. http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	RefreshInterval
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshOnLogOn

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshOnLogOn

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Set-CtxAppVServerSetting

March 11, 2024

Specifies the App-V Publishing Server settings to use on the VDA. These settings determine whether or not the App-V Client can automatically initiate a publishing refresh on certain events such as user logon or at specified intervals.

Syntax

```
Set-CtxAppVServerSetting
   -AppVPublishingServer <String>
    [-UserRefreshEnabled <Boolean>]
    [-UserRefreshOnLogon <Boolean>]
    [-UserRefreshInterval <Int32>]
    [-UserRefreshIntervalUnit <RefreshInterval>]
    [-GlobalRefreshEnabled <Boolean>]
    [-GlobalRefreshOnLogon <Boolean>]
    [-GlobalRefreshInterval <Int32>]
    [-GlobalRefreshIntervalUnit <RefreshInterval>]
    [<CommonParameters>]
```

Description

Specifies the App-V Publishing Server settings to use on the VDA. These settings determine whether or not the App-V Client can automatically initiate a publishing refresh on certain events such as user

logon or at specified intervals. Please refer to Microsoft App-V 5.0 documentation for more details on these settings : http://technet.microsoft.com/en-us/library/jj687745.aspx

Examples

EXAMPLE 1

```
Set-CtxAppVServerSetting -AppVPublishingServer http://appv-pubsrv.
    mydomain.com:8082 -GlobalRefreshEnabled $true -GlobalRefreshOnLogon
    $true -GlobalRefreshInterval 2 -GlobalRefreshIntervalUnit Hour -
    UserRefreshEnabled $true -UserRefreshOnLogon $true -
    UserRefreshInterval 3 -UserRefreshIntervalUnit Hour
```

Parameters

-AppVPublishingServer

The full URL (including port number) of the Publishing Server for which settings are to be set.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-UserRefreshEnabled

Enables a refresh of packages published to user groups either at user logon or at a specified interval. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to user groups on every user logon. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to user groups. This can be either days or hours, as specified by the UserRefreshIntervalUnit setting. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserRefreshIntervalUnit

Specifies the unit for the UserRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	RefreshInterval
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshEnabled

Enables a refresh of packages published to machine groups either at user logon or at a specified interval. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshOnLogon

Specifies whether or not to initiate a refresh of packages published to machine groups on every user logon. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshInterval

Specifies the frequency at which to initiate a refresh of packages published to machine groups. This can be either days or hours, as specified by the GlobalRefreshIntervalUnit setting. For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GlobalRefreshIntervalUnit

Specifies the unit for the GlobalRefreshInterval setting. This can be set to either Hours (0) or Days (1). For more information, see the App-V documentation on the Microsoft website at http://technet.microsoft.com/en-us/library/jj687745.aspx

Type:	RefreshInterval
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Test-CtxAppVServer

March 11, 2024

Tests the given URL for the presence of App-V Management and Publishing servers.

Syntax

```
Test-CtxAppVServer
    -AppVManagementServer <String>
    [<CommonParameters>]
```

```
Test-CtxAppVServer
    -AppVPublishingServer <String>
    [<CommonParameters>]
```

Description

Tests the given URL for the presence of App-V Management and Publishing Servers.

Examples

EXAMPLE 1

Tests whether “appv-mansrv”is a Management Server or not. The name can, but does not need to be specified as a Fully Qualified Domain Name (FQDN). No port number is required.

```
Test-CtxAppVServer -AppVManagementServer "appv-mansrv"
```

EXAMPLE 2

Tests whether “http://appv-pubsrv.mydomanin.com:8082”is a Publishing Server or not. Specify the full URL address, including FQDN and port number, of the Publishing Server.

```
Test-CtxAppVServer -AppVPublishingServer "http://appv-pubsrv.mydomanin.
    com:8082"
```

Parameters

-AppVManagementServer

Machine name of the App-V Management Server.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AppVPublishingServer

The URL (including the port number) of the App-V Publishing Server.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Related Links

about_Broker_AccessPolicy

March 11, 2024

Topic

Citrix Broker SDK - Access Policy

Short Description

Controls client-connection-based access to desktop groups.

Long Description

The site's access policy defines rules controlling a user's access to desktop groups. Access checks are based on details of the user's connection from their user device to the site. Think of the access policy informally as a connection-based firewall.

The access policy comprises a set of rules. Each rule:

- Relates to a single desktop group.
- Contains a set of connection filters and access right controls.

Multiple rules can apply to the same desktop group.

By default, users have no access to any desktop group within a site. A user gains access to a group when their connection's details match the connection filters of one or more rules in the access policy.

The access policy also grants control rights over desktop and application sessions. For example, it can specify which protocols are allowed for a connection from a given endpoint, and whether the user can restart their machine.

To use a resource published by a site, the user must have both access to the desktop group that contains it, and an entitlement to use the resource. Entitlements are typically granted by the site entitlement and assignment policies; see about_Broker_Policies for more information.

Access Policy Rules

A single access policy rule relates to a single specified desktop group and comprises a set of connection filters and access right grants as described below.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the access policy is evaluated.

Connection Filters Overview

The connection filters in an access policy rule comprise the following:

- Local/remote client (SmartAccess) filters
- Client IP address filters
- Client name filters

- User filters

All filters have an include and exclude form that can be individually enabled or disabled. For a rule to be considered when the access policy is evaluated, at least one connection include filter must be enabled. By default, all filters, both include and exclude, are disabled.

The detailed behavior of connection filters is covered later.

Access Right Controls Overview

The access right controls in an access policy rule comprise the following:

- Allowed protocols
- Whether machine restart, or programmatic session logoff, is allowed

The detailed behavior of access right controls is covered later.

Details Of Connection Filters

To gain access to a desktop group the user's connection must match the filter criteria of at least one access policy rule for that group.

To match a rule, a connection must match all the rule's enabled include connection filters and must not match any of the rule's enabled exclude filters. That is, entries in exclude filters take priority.

Because all rules are evaluated independently, if an exclude filter match prevents a connection gaining access to a desktop group through one rule, the connection may still gain access to the same group through a different rule.

The filters are described in pairs below, but within a single rule a match against any exclude filter prevents a connection from gaining access through that rule irrespective of which include filters within the rule were also matched.

Smartaccess Filters

SmartAccess filters allow filtering based on whether the client is directly connected (for example over a local area network (LAN)) or through Access Gateway. For connections through Access Gateway further filtering can be

performed based on the tags supplied from Access Gateway itself. The key properties of SmartAccess filters are:

- AllowedConnections (include filter: Filtered, NotViaAG, ViaAG)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  - Filtered (default)

    The filter matches any user connection not through Access Gateway. In addition, the filter may match user connections through Access Gateway subject to the following: if the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user's connection.

  - NotViaAG

    The filter matches only user connections not through Access Gateway. The contents of the IncludedSmartAccessTags property are ignored.

  - ViaAG

    The filter matches only user connections through Access Gateway. If the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user's connection.

  - AnyViaAG

    The filter matches only user connections through Access Gateway. The contents of the IncludedSmartAccessTags property are ignored. The behaviour of AnyViaAG is therefore the same as using ViaAG with an empty IncludedSmartAccessTags property.

  The IncludedSmartAccessTags property referred to above forms part of the include filter and is used if AllowedConnections is set to Filtered or ViaAG. It comprises a simple list of Access Gateway tags that are matched against those provided in the user's connection details.

- ExcludedSmartAccessTags (exclude filter)

  A simple list of Access Gateway tags that are matched against those provided in the user's connection details. If any tag in the list

matches one supplied with the user's connection, the user's connection does not match the access policy rule containing the filter.

The exclude filter has no setting corresponding to the AllowedConnections property so its behavior is determined only by the ExcludedSmartAccessTags property.

SmartAccess filters are typically used to control local (through a LAN) and remote (through Access Gateway) access to a site. A common model is to define two access policy rules rules for a group, one for local access and one for remote. The remote rule might impose restrictions on the user device having appropriate antivirus software installed, and potentially exclude certain user groups who would be allowed access over the corporate LAN (see USER FILTERS below).

Client Ip Filters

Client IP filters allow filtering based on the IP address of the user's device. The key properties of client IP filters are:

- IncludedClientIPs (include filter)

  A simple list of numeric IP address ranges that are matched against the user device. The filter matches if the device address falls within any of the ranges in the list.

- ExcludedClientIPs (exclude filter)

  A simple list of numeric IP address ranges that are matched against the user device. If any entry matches the device address, the user's connection does not match the access policy rule containing the filter.

An IP address range in these filters can be specified as a simple IP address or as a range using a conventional subnet mask.

Client Name Filters

Client name filters allow filtering based on the name of the user's device. The key properties of client name filters are:

- IncludedClientNames (include filter)

  A simple list of names that are matched against the user device. The filter matches if the device name matches any value in the list.

- ExcludedClientNames (exclude filter)

  A simple list of device names that are matched against the user device. If any entry matches the device name, the user's connection does not match the access policy rule containing the filter.

Note: The form of the device name presented to the site depends on the site configuration. For example, by default in these filters you cannot use the form of the name presented by Web Interface.

User Filters

User filters allow filtering based on the identity of the user. The key properties of user filters are:

- AllowedUsers (include filter: Filtered, AnyAuthenticated, Any)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  - Filtered (default)

    The filter matches if the user's logon token contains one or more users or user groups matching those specified in the IncludedUsers property. The IncludedUsers property is a simple list of users or user groups and is used only when the AllowedUsers property is set to Filtered.

  - AnyAuthenticated

    The filter matches any authenticated Microsoft Windows user. The contents of the IncludedUsers property are ignored.

  - Any

    The filter matches any user. The contents of the IncludedUsers property are ignored. In the current implementation this value is handled in the same way as AnyAuthenticated.

- ExcludedUsers (exclude filter)

  A simple list of users or user groups. If any entry matches one in the user's logon token, the user's connection does not match the access policy rule containing the filter.

  The exclude filter has no setting corresponding to the AllowedUsers property so its behavior is determined only by the ExcludedUsers property.

Details Of Access Right Controls

The access right controls of an access policy rule determine rights that the user has over any desktop or application session that they obtain from the rule's desktop group.

The rights apply only if the user's connection matches the connection filters of a rule, and only if the user also has an entitlement to a desktop or application session from the associated desktop group.

The following properties define the access rights:

- AllowedProtocols

  A simple list of communication protocols over which connections can be made to resources published by the desktop group. For example, use this to restrict protocols with high bandwidth requirements to connections originating from a LAN.

- AllowRestart

  For single-session power-managed machines, allows the user to restart the machine (the machine is powered off using the capabilities of its hypervisor). For multi-session machines the user's session is simply logged off.

For a given connection, if multiple rules result in access being granted to a session from a desktop group, the user's rights are the combined rights of all the rules that matched for that group. The allowed protocol lists from all the rules are combined, and the user is granted restart rights if any one rule has AllowRestart set.

See Also

- about_Broker_Policies
- about_Broker_AssignmentPolicy
- about_Broker_EntitlementPolicy
- New-BrokerAccessPolicyRule
- Get-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule
- New-BrokerAssignmentPolicyRule
- New-BrokerEntitlementPolicyRule

- New-BrokerAppAssignmentPolicyRule
- New-BrokerAppEntitlementPolicyRule
- Add-BrokerUser

about_Broker_AccessPolicy

March 11, 2024

Topic

Citrix Broker SDK - Access Policy

Short Description

Controls client-connection-based access to desktop groups.

Long Description

The site's access policy defines rules controlling a user's access to desktop groups. Access checks are based on details of the user's connection from their user device to the site. Think of the access policy informally as a connection-based firewall.

The access policy comprises a set of rules. Each rule:

- Relates to a single desktop group.
- Contains a set of connection filters and access right controls.

Multiple rules can apply to the same desktop group.

By default, users have no access to any desktop group within a site. A user gains access to a group when their connection's details match the connection filters of one or more rules in the access policy.

The access policy also grants control rights over desktop and application sessions. For example, it can specify which protocols are allowed for a connection from a given endpoint, and whether the user can restart their machine.

To use a resource published by a site, the user must have both access to the desktop group that contains it, and an entitlement to use the resource. Entitlements are typically granted by the site entitlement and assignment policies; see about_Broker_Policies for more information.

Access Policy Rules

A single access policy rule relates to a single specified desktop group and comprises a set of connection filters and access right grants as described below.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the access policy is evaluated.

Connection Filters Overview

The connection filters in an access policy rule comprise the following:

- Local/remote client (SmartAccess) filters
- Client IP address filters
- Client name filters
- User filters

All filters have an include and exclude form that can be individually enabled or disabled. For a rule to be considered when the access policy is evaluated, at least one connection include filter must be enabled. By default, all filters, both include and exclude, are disabled.

The detailed behavior of connection filters is covered later.

Access Right Controls Overview

The access right controls in an access policy rule comprise the following:

- Allowed protocols
- Whether machine restart, or programmatic session logoff, is allowed

The detailed behavior of access right controls is covered later.

Details Of Connection Filters

To gain access to a desktop group the user's connection must match the filter criteria of at least one access policy rule for that group.

To match a rule, a connection must match all the rule's enabled include connection filters and must not match any of the rule's enabled exclude filters. That is, entries in exclude filters take priority.

Because all rules are evaluated independently, if an exclude filter match prevents a connection gaining access to a desktop group through one rule, the connection may still gain access to the same group through a different rule.

The filters are described in pairs below, but within a single rule a match against any exclude filter prevents a connection from gaining access through that rule irrespective of which include filters within the rule were also matched.

Smartaccess Filters

SmartAccess filters allow filtering based on whether the client is directly connected (for example over a local area network (LAN)) or through Access Gateway. For connections through Access Gateway further filtering can be performed based on the tags supplied from Access Gateway itself. The key properties of SmartAccess filters are:

- AllowedConnections (include filter: Filtered, NotViaAG, ViaAG)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  - Filtered (default)

    The filter matches any user connection not through Access Gateway. In addition, the filter may match user connections through Access Gateway subject to the following: if the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user's connection.

  - NotViaAG

The filter matches only user connections not through Access Gateway. The contents of the IncludedSmartAccessTags property are ignored.

- ViaAG

  The filter matches only user connections through Access Gateway. If the IncludedSmartAccessTags property is empty, any such connection matches. However, if the property is not empty at least one SmartAccess tag from the filter property must match a SmartAccess tag supplied with the user's connection.

- AnyViaAG

  The filter matches only user connections through Access Gateway. The contents of the IncludedSmartAccessTags property are ignored. The behaviour of AnyViaAG is therefore the same as using ViaAG with an empty IncludedSmartAccessTags property.

The IncludedSmartAccessTags property referred to above forms part of the include filter and is used if AllowedConnections is set to Filtered or ViaAG. It comprises a simple list of Access Gateway tags that are matched against those provided in the user's connection details.

- ExcludedSmartAccessTags (exclude filter)

  A simple list of Access Gateway tags that are matched against those provided in the user's connection details. If any tag in the list matches one supplied with the user's connection, the user's connection does not match the access policy rule containing the filter.

  The exclude filter has no setting corresponding to the AllowedConnections property so its behavior is determined only by the ExcludedSmartAccessTags property.

SmartAccess filters are typically used to control local (through a LAN) and remote (through Access Gateway) access to a site. A common model is to define two access policy rules rules for a group, one for local access and one for remote. The remote rule might impose restrictions on the user device having appropriate antivirus software installed, and potentially exclude certain user groups who would be allowed access over the corporate LAN (see USER FILTERS below).

Client Ip Filters

Client IP filters allow filtering based on the IP address of the user's device. The key properties of client IP filters are:

- IncludedClientIPs (include filter)

  A simple list of numeric IP address ranges that are matched against the user device. The filter matches if the device address falls within any of the ranges in the list.

- ExcludedClientIPs (exclude filter)

  A simple list of numeric IP address ranges that are matched against the user device. If any entry matches the device address, the user's connection does not match the access policy rule containing the filter.

An IP address range in these filters can be specified as a simple IP address or as a range using a conventional subnet mask.

Client Name Filters

Client name filters allow filtering based on the name of the user's device. The key properties of client name filters are:

- IncludedClientNames (include filter)

  A simple list of names that are matched against the user device. The filter matches if the device name matches any value in the list.

- ExcludedClientNames (exclude filter)

  A simple list of device names that are matched against the user device. If any entry matches the device name, the user's connection does not match the access policy rule containing the filter.

Note: The form of the device name presented to the site depends on the site configuration. For example, by default in these filters you cannot use the form of the name presented by Web Interface.

User Filters

User filters allow filtering based on the identity of the user. The key properties of user filters are:

- AllowedUsers (include filter: Filtered, AnyAuthenticated, Any)

  This property controls the behavior of the include filter. The default value is Filtered. The possible values are as follows:

  - Filtered (default)

    The filter matches if the user's logon token contains one or more users or user groups matching those specified in the IncludedUsers property. The IncludedUsers property is a simple list of users or user groups and is used only when the AllowedUsers property is set to Filtered.

  - AnyAuthenticated

    The filter matches any authenticated Microsoft Windows user. The contents of the IncludedUsers property are ignored.

  - Any

    The filter matches any user. The contents of the IncludedUsers property are ignored. In the current implementation this value is handled in the same way as AnyAuthenticated.

- ExcludedUsers (exclude filter)

  A simple list of users or user groups. If any entry matches one in the user's logon token, the user's connection does not match the access policy rule containing the filter.

  The exclude filter has no setting corresponding to the AllowedUsers property so its behavior is determined only by the ExcludedUsers property.

Details Of Access Right Controls

The access right controls of an access policy rule determine rights that the user has over any desktop or application session that they obtain from the rule's desktop group.

The rights apply only if the user's connection matches the connection filters of a rule, and only if the user also has an entitlement to a desktop or application session from the associated desktop group.

The following properties define the access rights:

- AllowedProtocols

A simple list of communication protocols over which connections can be made to resources published by the desktop group. For example, use this to restrict protocols with high bandwidth requirements to connections originating from a LAN.

- AllowRestart

  For single-session power-managed machines, allows the user to restart the machine (the machine is powered off using the capabilities of its hypervisor). For multi-session machines the user's session is simply logged off.

For a given connection, if multiple rules result in access being granted to a session from a desktop group, the user's rights are the combined rights of all the rules that matched for that group. The allowed protocol lists from all the rules are combined, and the user is granted restart rights if any one rule has AllowRestart set.

See Also

- about_Broker_Policies
- about_Broker_AssignmentPolicy
- about_Broker_EntitlementPolicy
- New-BrokerAccessPolicyRule
- Get-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule
- New-BrokerAssignmentPolicyRule
- New-BrokerEntitlementPolicyRule
- New-BrokerAppAssignmentPolicyRule
- New-BrokerAppEntitlementPolicyRule
- Add-BrokerUser

about_Broker_Applications

March 11, 2024

Topic

Citrix Broker SDK - Applications

Short Description

Describes how to publish and manage hosted applications.

Long Description

Published applications allow your users to launch applications as if they were installed on their devices when in fact they are hosted remotely. The applications are normally launched in a seamless window, meaning that users see only the application window and not an additional desktop.

Published applications are hosted on either desktop operating systems or server operating systems. Applications are published to users and desktop groups. As such, conceptually they exist "on top"of desktop groups, which are themselves built on top of catalogs. See about_Broker_Concepts for more information on catalogs.

There are three types of applications:

- HostedOnDesktop - application runs on a remote machine and is displayed on the local client desktop.
- InstalledOnClient - application is installed and run on the local client machine and has its window overlaid on to a remote desktop.
- PublishedContent - application refers to actual content (using a URL or UNC). File type associations determine what application is run to view the content

Hosting Applications

There are three main ways of hosting an application: using a private single-session VDI desktop, a shared single-session VDI desktop, and on shared multi-session server operating systems.

An application hosted on a shared single-session VDI desktop, when launched, is hosted on a random machine within the desktop group. An application hosted on a private single-session VDI desktop ensures that when a user launches the application it is always hosted on the same machine. An

application hosted on shared multi-session server operating systems ensures that when a user launches the application it is always hosted on one of the least loaded servers in the desktop group.

You control how the application is hosted based on the kind of desktop group that you choose. Shared desktop groups randomly select a machine, and these desktop groups can only be created from catalogs with a Random AllocationType. On the other hand, private desktop groups host the application on the same machine for that user every time, and these desktop group types can only be created from catalogs with a Permanent AllocationType.

User Access & Assignment

Users are not assigned an application directly, but instead they are required to first have access to the desktop groups on which the applications are published through access policy rules.

With shared desktop groups, access to a published application also needs an application entitlement policy rule. An application entitlement policy rule defines the set of users who are allowed per-session access to desktops in a specified desktop group.

With private desktop groups, access to a published application also needs an application assignment policy rule. An application assignment policy rule defines the set of users who are allowed access to a single application session in a desktop group.

Users are assigned private machines in one of two ways: pre-assigned or assign-on-first-use. Pre-assigned means that individual user accounts have been specified for the machines within that desktop group. A single machine can only have a single user account (not a group account) assigned to it, and likewise a user can only be assigned a single machine within a desktop group.

Assign-on-first-use requires less configuration. Machines are assigned to users the first time they log on. Rather than allocating users directly to machines, instead users are assigned to the private desktop group, either through individual user accounts or through user group accounts. Then, when a user assigned to the desktop group logs on, a machine is automatically allocated to them.

User Visibility

Users can be further filtered by configuring the visibility filter on top of the application. This restricts the application to a subset of the users that were granted access by the access policy and entitlement/assignment policy rules on the desktop group.

Multiple Desktop Groups

You can publish an application to multiple desktop groups, which have to be of the same kind. Generally, an application that is published only to private desktop groups is referred to as a "private application". An application that is only published to shared desktop groups is referred to as a "shared application".

Application Groups

An application group is a group of applications. Applications may be published either by adding them directly to a desktop group, or they may first be assigned to an application group, which may then be added to a desktop group. Adding an application group to a desktop group publishes all of the application group's applications to that desktop group in a single operation. The set of machines on which the application group's applications can be launched may then be restricted further by tagging the desired machines and setting the application group's RestrictToTag property.

Grouping applications together allows for them to be administered as a single unit. For example:

- Moving an application group to a different desktop group has the same effect as moving each of the applications, without the need to move each application individually.
- Disabling an application group disables each application in the group.
- Application groups have a visibility filter that applies to every application within the group (see 'User visibility' above).

Application groups may be configured to allow or disallow session sharing with applications published by other application groups. For example, you may have a particular set of applications that do not interoperate well with certain other applications that are installed on the same machines - for example, two different versions of the same software suite, or two different

versions of the same web browser. You may prefer not to allow the user to launch both versions in the same session.

In such a case, an application group can be created for each version of the software suite, and the applications for each version of the software suite added to the corresponding application group. If session sharing is disabled for each of the application groups, then the user will be able to run applications of the same version in the same session, and can still run other applications at the same time - but not in the same session. If the user launches one of the different-versioned applications, which are in a different application group, or launches any application that is not contained in an application group, then that application is launched in a new session.

IMPORTANT. The session sharing feature of application groups is not intended to be a security sandboxing feature. In addition to not being foolproof, it cannot prevent users from launching applications into their sessions through other means (e.g. through Windows Explorer).

An application may be a member of more than one application group. In that case, it inherits settings from all of the application groups of which it is a member. For example, an application is only disabled if all of the application groups to which it is assigned are disabled, or if the application itself is disabled.

However, if application or application group visiblity filters are used, then it is recommended that each application be a member of at most one application group, and that visibility filters are used either on applications, or on application groups, but not both. This is because it is not generally possible for the system to combine multiple visibility filters in such as way that they can consistently be enforced.

An application may also be a member of both an application group and a desktop group at the same time. However, this is not recommended, because the additional complexity can make it harder for the administrator to predict how machines are allocated to users. It is recommended that each application be published either via application groups, or via desktop groups, but not both.

Application groups may not be published via private desktop groups.

Application groups are purely an administrative concept. They are not visible to end users.

Licensing

Hosted applications need the appropriate licenses to exist on the license server to be functional.

Naming

Applications have three names that identify them: the Name, BrowserName and the PublishedName. The Name is unique for each application, is not visible to the user, and is primarily used for administrative purpose. The BrowserName is unique across the entire site, and is primarily used internally. The PublishedName is not unique and is the name seen by end users who have access to this application.

When creating an application, you only need to specify the Name. If no BrowserName is specified one is automatically generated. If no PublishedName is specified by default it is the same as the Name.

The following special characters are not allowed in the Name, BrowserName or the PublishedName properties: \ / ; : # . * ? = < > | [ ] ( ) ‘“

In addition the ‘ character is not allowed in the Name property.

To change the PublishedName or BrowserName of an application you must use the Set-BrokerApplication cmdlet.

To change the Name of an application you must use the Rename-BrokerApplication cmdlet.

Optional

You can configure file-type associations for applications, so that if a user double-clicks a document icon on their device, a published application automatically starts. For more information, see the help for New-BrokerConfiguredFTA.

You can apply tags to applications as another convenient way to further organize (and search for) applications. For more information, see the help for New-BrokerTag.

Cmdlets

New-BrokerApplication

Creates an application for publishing after the needed desktop groups, access policy and entitlement/assignment policy rules have been created.

Add-BrokerApplication

Adds one or more applications to a desktop group.

Get-BrokerApplication

Gets one or more applications.

Remove-BrokerApplication

Deletes one or more applications or it can be used to delete just the association of an application to a desktop group.

Rename-BrokerApplication

Changes the Name of an application.

Set-BrokerApplication

Changes the settings of application objects, except their names.

Examples

Shared Application (Singlesession)

You have created a catalog with a Random AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with shared hosting as follows:

```
Write-Host "Create a desktop group, and add machines to it"
$dg = New-BrokerDesktopGroup "Shared Application Group"
-DesktopKind Shared -DeliveryType AppsOnly -SessionSupport '
    SingleSession'
$m1 = Get-BrokerMachine -MachineName "ACME\worker1"
$m2 = Get-BrokerMachine -MachineName "ACME\worker2"
Add-BrokerMachine $m1 -DesktopGroup $dg
Add-BrokerMachine $m2 -DesktopGroup $dg
Write-Host "Create access policy rule for desktop group"
New-BrokerAccessPolicyRule -Name "Shared Application Group - Allow
    Everyone Access"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled
    $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
Write-Host "Create application entitlement policy for desktop group"
New-BrokerAppEntitlementPolicyRule -Name "Shared Application Group -
    App Entitlement"
-IncludedUsers 'ACME\Domain Users' -DesktopGroupUid $dg.Uid -Enabled
    $true
```



```
Write-Host "Create an application"
New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
-CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid
```

In turn, this set of commands: creates a shared single session desktop group for applications delivery; adds two machines (from a catalog with a Random AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy and application entitlement policy rules; creates an application; specifies its name, the executable, and links the application to the desktop group that will host it.

Shared Application (Multisession)

You have created a catalog with a Random AllocationType and MultiSession SessionSupport. It contains one machine called worker1 in the ACME domain. You publish an application with shared hosting as follows:

```
Write-Host "Create a desktop group, and add machines to it"
$dg = New-BrokerDesktopGroup "Shared Application Group"
-DesktopKind Shared -DeliveryType AppsOnly -SessionSupport '
    MultiSession'
$m1 = Get-BrokerMachine -MachineName "ACME\worker1"
Add-BrokerMachine $m1 -DesktopGroup $dg
Write-Host "Create access policy rule for desktop group"
New-BrokerAccessPolicyRule -Name "Shared Application Group - Allow
    Everyone Access"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled
    $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
Write-Host "Create application entitlement policy for desktop group"
New-BrokerAppEntitlementPolicyRule -Name "Shared Application Group -
    App Entitlement"
-IncludedUsers 'ACME\Domain Users' -DesktopGroupUid $dg.Uid -Enabled
    $true
Write-Host "Create an application"
New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
-CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid
```

In turn, this set of commands: creates a shared multi session desktop group for applications delivery; adds one machine (from a catalog with a Random AllocationType and MultiSession SessionSupport) to the desktop group; creates access policy and application entitlement policy rules; creates an application; specifies its name, the executable, and links the application to the desktop group that will host it.

Private Pre-Assigned Application

You have a catalog with a Permanent AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with private hosting using pre-assigned machines as follows:

```
Write-Host "Create a desktop group, and add machines to it"
$dg = New-BrokerDesktopGroup "Private App G1" -DesktopKind Private
-DeliveryType AppsOnly -SessionSupport 'SingleSession'
$m1 = Get-BrokerMachine -MachineName "ACME\worker1"
$m2 = Get-BrokerMachine -MachineName "ACME\worker2"
Add-BrokerMachine $m1 -DesktopGroup $dg
Add-BrokerMachine $m2 -DesktopGroup $dg
Write-Host "Setting access policy rule for desktop group"
New-BrokerAccessPolicyRule -Name "Private App G1 - Allow Everyone
    Access"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled
    $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
Write-Host "Pre-Assign users to the machines"
Add-BrokerUser "ACME\user1" -Machine $m1
Add-BrokerUser "ACME\user2" -Machine $m2
Write-Host "Create an application"
New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
-CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid
```

In turn, this set of commands: creates a private desktop group for applications delivery; adds two machines (from a catalog with a Permanent AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy; creates two user objects; pre-assigns a user to each machine; creates the application; assigns users to it; and links the application to the desktop group that will host it.

Private, Assign-On-First-Use Application

You have a catalog created with a Permanent AllocationType and SingleSession SessionSupport. It contains two machines called worker1 and worker2, both in the ACME domain. You publish an application with private hosting using assign-on-first-use machines as follows:

```
Write-Host "Create a desktop group, and add machines to it"
$dg = New-BrokerDesktopGroup "Private App G2" -DesktopKind Private
-DeliveryType AppsOnly -SessionSupport 'SingleSession'
$m1 = Get-BrokerMachine -MachineName "ACME\worker1"
$m2 = Get-BrokerMachine -MachineName "ACME\worker2"
Add-BrokerMachine $m1 -DesktopGroup $dg
```



```
Add-BrokerMachine $m2 -DesktopGroup $dg
Write-Host "Setting access policy rule for desktop group"
New-BrokerAccessPolicyRule -Name "Private App G1 - Allow Everyone
    Access"
-Enabled $true -DesktopGroupUid $dg.Uid -IncludedUserFilterEnabled
    $true
-AllowedProtocols @("HDX") -AllowedUsers AnyAuthenticated
Write-Host "Create application assignment policy for desktop group"
New-BrokerAppAssignmentPolicyRule -Name "Private App G2 - App
    Assignment"
-IncludedUsers 'ACME\Domain Users' -DesktopGroupUid $dg.Uid -Enabled
    $true
Write-Host "Create an application"
New-BrokerApplication -Name "Notepad" -PublishedName "Notepad"
-CommandLineExecutable "notepad.exe" -DesktopGroup $dg.Uid
```

In turn, this set of commands: creates a private single session desktop group for applications delivery; adds two machines (from a catalog with a Permanent AllocationType and SingleSession SessionSupport) to the desktop group; creates access policy and application assignment policy rules; creates the application, and links the application to the desktop group that will host it.

See Also

- about_Broker_Concepts
- about_Broker_Desktops
- New-BrokerApplication
- Add-BrokerApplication
- Remove-BrokerApplication
- Rename-BrokerApplication
- Set-BrokerApplication
- New-BrokerAppAssignmentPolicyRule
- Remove-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- New-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule

about_Broker_AssignmentPolicy

March 11, 2024

Topic

Citrix Broker SDK - Assignment Policy

Short Description

Controls the automatic, permanent assignment of machines to users.

Long Description

The site's assignment policy defines rules controlling automatic assignment of machines to users in a process referred to as Assign On First Use (AOFU).

In this assignment model, a desktop group is initially populated with machines that have no assignments, and users are granted entitlements to obtain a machine selected at random from the pool by self-service assignment. Once made, the assignment is permanent. The resulting assigned machines can be used to deliver either desktop or application sessions depending on the delivery type of the desktop group.

The assignment policy comprises a set of rules. The policy can be applied only to desktop groups of desktop kind Private.

For assignment of machines to deliver desktop sessions:

- Multiple rules can apply to the same desktop group.
- Each rule grants an entitlement to one or more machines.

For assignment of machines to deliver application sessions:

- Only a single rule can apply to a given desktop group.
- Each rule grants an entitlement to a single machine.
- Although only a single application assignment rule can be defined for a group, a user can still launch multiple applications from that group because the applications all run within the same session on the assigned machine.

Once a machine is assigned to a user by an assignment policy rule, the rule plays no further part in controlling access to that machine. The rule can be changed, or even removed, without impacting the user's access to the machine.

Rules for desktop and application machine assignments are distinct. Desktop assignments are managed through the BrokerAssignmentPolicyRule SDK object, and application rules through the BrokerAppAssignmentPolicyRule object.

Desktop machine assignment rules can be created only for desktop groups with delivery type DesktopsOnly, whereas an application machine assignment rule can be created only for delivery type AppsOnly.

Because desktop groups of assigned machines do not allow delivery type DesktopsAndApps, desktop machine assignment and application machine assignment rules cannot exist for the same desktop group.

Assignment policy rules are also used to configure the RemotePC feature where their detailed usage differs. For more information on the RemotePC feature see "help about_Broker_RemotePC".

For an entitlement granted by the assignment policy to be available to a user, the site's access policy must also grant them access to the desktop group.

Assignment Policy Rules

Each assignment policy rule has the following key properties:

- The desktop group to which it applies (one rule only ever applies to one group).
- The users to which machines can be assigned by the rule.

Additionally for desktop assignment rules, the following properties exist:

- The published name of the entitlement (visible to the user).
- The number of machines (entitlements) to which the rule grants access.
- The properties that a newly assigned desktop acquires.

If multiple desktop assignment rules entitle a user to machines from the same desktop group, their total machine entitlement is the sum of those granted by all those rules. The properties of the desktop sessions obtained may differ depending on which of the entitlements the user selects to start a session.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the assignment policy is evaluated.

User Filters (Full)

Each rule has two user filters, an include filter and an exclude filter:

- The include filter contains users and user groups that are granted entitlements to machines.
- The exclude filter contains users and user groups that are denied entitlements to machines.

If the include filter of a rule contains multiple instances of a user (either explicit or implicit), this does not alter the number of machine entitlements granted to them by the rule.

Entries in the exclude filter take priority, so if a user appears explicitly or implicitly in both filters, access is denied. Typically, you use this filter to exclude a user or group of users who would otherwise gain access because they are members of a user group specified in the include filter.

Because all rules are independently evaluated, the exclude filter can exclude only users who would otherwise gain an entitlement through the same rule's include filter. That is, if a user is in a rule's include filter but not its exclude filter, the rule is guaranteed to grant that user access to a machine irrespective of whether the user appears in the exclude filter of other rules.

If a filter contains a user group that contains other users and groups, the filter implicitly includes all of those users and groups.

By default the exclude filter is disabled.

To maintain assignment policy rules that can be fully displayed and edited with Citrix Studio, use the simplified user filter model below and do not use the exclude filter.

User Filters (Simplified)

The included user filter described above also supports a simplified usage model where the filter itself is disabled. When this is done, any user who has access to the desktop group through the access policy is implicitly granted an entitlement to a machine through the entitlement policy rule without the need to list the user in the rule's include filter.

This is useful in cases where the access policy for the desktop group already explicitly specifies the users who should have access.

Even if the include filter is disabled, the exclude filter can still be used to deny the entitlement to users who would otherwise gain access through the access policy alone.

This simplified usage cannot be used for RemotePC desktop groups.

Remote Pc Usage

When a desktop group is configured as a RemotePC group, the usage of the assignment policy differs in the following ways:

- Only a single rule can apply to a given desktop group.
- The delivery type of the desktop group must be DesktopsOnly.
- The simplified user filter model cannot be used. Users or user groups must appear explicitly in the included user filter of the assignment policy rule.
- If the included user filter is disabled then RemotePC assignment for the group is also effectively disabled.

For more information on the RemotePC feature see "help about_Broker_RemotePC".

Additional Desktop Assignment Rule Properties

Desktop assignment rules specify the following additional properties:

- PublishedName
- Description
- IconUid
- ColorDepth
- SecureIcaRequired

The published name, description, and icon UID properties apply to the desktop entitlement itself and determine the way in which the entitlement is presented to the user in, for example, StoreFront.

The color depth and secure ICA properties apply to the desktop session that is obtained when the entitlement is used.

In all cases, these properties can be explicitly specified. However, a null value (the default) means that the corresponding property is taken from the desktop group to which the rule applies. This inheritance from groups is

dynamic; if the property of the group changes, the property of the entitlement changes too.

For assignment rules these properties are copied to the newly assigned desktop when the granted entitlement is first used. If the properties on the rule are subsequently changed the properties on the user's assigned desktop do not change. The dynamic inheritance of desktop properties from the desktop group continues after the assignment has occurred if the original rule did not provide explicit property values.

Calculating Overall Machine Entitlements

Assignment rules are modified only by the SDK; they are not modified when used by the system to make automatic assignments. The number of machine entitlements offered to the user is determined by the number of machines (within the desktop group) already assigned to them and by which rules those assignments were made.

For each desktop group, the number of desktop entitlements available to the user is determined as follows:

1. The total number of entitlements for the user to machines in the group, from all rules, is calculated.
2. The total number of assigned machines (from any source) that the user already has in the group is determined.
3. The outstanding entitlement value is derived as the difference of the above two numbers.
4. If the outstanding entitlement value is zero or fewer, no further entitlements are allowed.
   Otherwise, for each applicable rule, the number of entitlements is that defined by the rule itself, minus the number of desktops already assigned to the user by that rule, and capped by the outstanding entitlement value.

Desktop and application machine entitlements both follow the above rules. However, because only a single application assignment rule granting a single entitlement per group can exist, these rules are seldom a consideration for applications.

Examples

- Simple case:
  A user is entitled to a single machine in a group by a single assignment rule:

  1. On first use, the user sees a single desktop entitlement.
     If the user selects this, a new desktop is assigned.
  2. On subsequent uses, the user sees the assigned desktop only.
     No other entitlements are displayed.

- Complex case:
  A user is entitled to two machines, one from each of two different rules, A and B, both applying to the same desktop group. In addition, the user has a machine explicitly assigned to them by the administrator within that group:

  1. On first use, the user sees the administrator-assigned desktop, a single entitlement to a desktop of type A, and a single entitlement to a desktop of type B. If either of the entitlements is selected, a desktop of the appropriate type, A or B, is assigned.
  2. On subsequent uses, the user sees the administrator-assigned desktop and the desktop assigned by the selected entitlement.
     No further entitlements are displayed.

Machine Entitlement Presentation

For desktop machine assignment rules, the user interface determines whether users can visually distinguish between an entitlement to a machine and an actual assigned desktop. This cannot be controlled by Broker SDK cmdlets.

Although the number of entitlements seen by users takes account of all of their currently assigned desktops, the dynamic state of the system can affect the user interaction. This means that entitlements are displayed even where the pool of machines is empty, or the remaining desktops are in maintenance mode or otherwise unavailable. If the user attempts to use an entitlement in these cases, they receive a no-available-desktop error.

For application machine entitlement rules, the entitlements themselves are not presented to the end user; only the applications available to the user are presented. The use of available assignments, or of already assigned machines, is managed automatically.

Notes

If an assignment rule grants entitlements to a user group:

- The machine is assigned to the user who selects the entitlement, thus an assignment policy rule cannot be used to assign a machine to a user group. However, an administrator can assign a machine to a user group using the Add-BrokerUser cmdlet.
- The number of machine entitlements specified in a desktop assignment rule applies to each member of the user group. For example, if a rule grants a user group access to two machines, every user in the group is entitled to two desktops.

The total number of machine entitlements defined by the assignment policy for a given desktop group may exceed the number of machines present in the group. A user attempting to use an entitlement when the pool of machines is empty receives a no-desktop-available error. (See CALCULATING OVERALL MACHINE ENTITLEMENTS above).

A machine assigned to a user as a result of the assignment policy remains permanently assigned unless the machine assignment itself is removed by the administrator. Such assignments are permanent even if the assignment rule is deleted, when the machine is treated as administrator-assigned.

The assignment policy cannot be used to assign a machine to more than one user. This is only possible using the Add-BrokerUser cmdlet.

See Also

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_Applications
- New-BrokerAssignmentPolicyRule
- Get-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule
- Remove-BrokerAssignmentPolicyRule
- New-BrokerAppAssignmentPolicyRule
- Get-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule

- Remove-BrokerAppAssignmentPolicyRule
- Add-BrokerUser

about_Broker_AutoTagRule

March 11, 2024

Topic

Citrix Broker SDK - AutoTagRules

Short Description

Overview of administrator defined rules for automatic addition and removal of tags on machines, application groups, applications, desktop groups or desktops.

Long Description

Auto tag rules enable administrators to automatically to set and/or remove tags for machines, application groups, applications, desktop groups or desktops. The rules are periodically evaluated and the tags of associated objects updated.
Each auto tag rule is associated with exactly one Tag. A Tag is associated with at most one auto tag rule.

For example:
tag = New-BrokerTag -Name MyExample
New-BrokerAutoTagRule -Name StaticCatalogs -TagUid tag.Uid -ObjectType Catalog -RuleText "-AllocationType Static"

See Also

- New-BrokerAutoTagRule
- Get-BrokerAutoTagRule
- Set-BrokerAutoTagRule
- Rename-BrokerAutoTagRule
- Remove-BrokerAutoTagRule
- Add-BrokerTag

- Set-BrokerTag
- Remove-BrokerTag

about_Broker_Concepts

March 11, 2024

Topic

Citrix Broker - Concepts

Short Description

Overview of the Citrix Broker.

Long Description

The Citrix Broker is a Microsoft Windows service running on a delivery controller that responds to desktop/application launch requests from users through StoreFront by selecting a suitable machine, powering it up if necessary, and then returning the address of the selected machine to the user's endpoint system so that a session connection can be made. If required for resilience or scale, additional instances of the Broker may be installed to run on additional delivery controllers in the same delivery site.

In addition to handling launch requests, the Broker also has background responsibilities in the delivery site; these include: maintaining appropriate numbers of unused, powered-up machines to avoid unnecessary delays to users launching desktops/applications; maintaining periodic contact with powered-up machines; and monitoring the state of machines and user sessions.

The Citrix.Broker.Admin PowerShell snap-in provides the cmdlets needed to administer and monitor the behaviour of the Broker service, either on the local system (by default) or on another system (by use of the -AdminAddress command-line parameter).

The cmdlets in the broker SDK manage objects in the following broad functional areas:

Provisioning Configuration

The Broker must be informed of the machines which are at its disposal. In order to do this, machines are organized in catalogs, created with the New-BrokerCatalog cmdlet; machines are introduced into the system through the use of the New-BrokerMachine cmdlet.

Catalogs define the nature of the machines contained within them, such as the allocation type (that is, static or random), how the machines are actually provisioned (that is, by MCS, PVS or manually), whether they are physical or virtual machines, whether they are single-session or multi-session, etc.

Typically, machines configured from a provisioning standpoint are not associated with specific users (though they may need to be, for example if the machine's disk image was captured from a specific user's physical desktop using a P2V tool); this is normally done when configuring how resources are brokered to users, below.

It is also possible for a catalog to be configured to be populated automatically with end users' existing physical machines using the RemotePC feature. The about_Broker_RemotePC topic gives more detail.

Provisioning configuration involves the following SDK objects:

- BrokerHypervisorConnection
- BrokerCatalog
- BrokerMachine
- BrokerRemotePCAccount
- BrokerUser

For more information, see:

about_Broker_Machines
about_Broker_RemotePC
Get-BrokerHypervisorConnection
Get-BrokerCatalog
Get-BrokerMachine
Get-BrokerUser

Brokering Configuration

In order that resources (that is, desktops and applications) can be used in user sessions, the Broker must be configured to connect incoming user launch

requests through StoreFront with the correct machine. This is achieved by adding machines to desktop groups. The grouping of machines in desktop groups need not necessarily match the grouping of the machines within the catalogs that were used for the configuration of provisioning. It is through the desktop group that the configuration of which users can use which machine resources is achieved.

Configuration of the mapping between resources and end users is achieved through a combination of machine assignment and entitlement rules. In addition, access to those resources must also be configured (for example, some resources could be configured only to be accessible to users when they are not connecting remotely through Access Gateway.)
The about_Broker_Policies topic gives more detail of the rich configuration options available.

It is also possible for a desktop group to be configured to be populated automatically with end users existing physical machines using the RemotePC feature. The about_Broker_RemotePC topic gives more detail.

When machines are virtual, the broker can be configured to minimize power usage by switching them off when they are not expected to be in use while still maintaining good response times for end-user launch requests. This is achieved through power policy for desktop groups. This allows separate configuration for peak compared to off-peak times of the week of the number of machines nominally to be powered up, the number of machines to be powered up and idling, in addition to those in use to be used as a "buffer"to ensure prompt servicing of user launch requests, and the behavior required for virtual machines when users disconnect from their sessions for extended periods of time.

It is also possible to issue explicit power commands to machines. The about_Broker_PowerManagement topic gives more detail.

Configuration of Brokering involves the following SDK objects:

- BrokerDesktopGroup
- BrokerPrivateDesktop
- BrokerSharedDesktop
- BrokerRemotePCAccount
- BrokerPowerTimeScheme
- BrokerUser
- BrokerTag
- BrokerAccessPolicyRule

- BrokerAssignmentPolicyRule
- BrokerEntitlementPolicyRule
- BrokerApplication
- BrokerApplicationGroup
- BrokerApplicationInstance
- BrokerAppAssignmentPolicyRule
- BrokerAppEntitlementPolicyRule
- BrokerConfiguredFTA
- BrokerImportedFTA

For more information, see:

about_Broker_Desktops
about_Broker_Policies
about_Broker_Applications
about_Broker_RemotePC
Get-BrokerPrivateDesktop
Get-BrokerSharedDesktop
Get-BrokerPowerTimeScheme
Get-BrokerUser
Get-BrokerTag
Get-BrokerAccessPolicyRule
Get-BrokerAssignmentPolicyRule
Get-BrokerEntitlementPolicyRule

Monitoring And Administration

After you have provisioned and configured machines for brokering, use the broker SDK to monitor and administer user sessions and other aspects of the delivery site.

Monitoring and administration involve the following SDK objects:

- BrokerServiceStatus
- BrokerHypervisorAlert
- BrokerDesktop
- BrokerDesktopUsage
- BrokerHostingPowerAction
- BrokerSession
- BrokerSessionMessage

For more information, see:

about_Broker_Desktops
Get-BrokerServiceStatus
Get-BrokerHypervisorAlert
Get-BrokerDesktop
Get-BrokerDesktopUsage
Get-BrokerHostingPowerAction
Get-BrokerSession
Send-BrokerSessionMessage

Site Management

The broker must be configured after installation; this is normally performed automatically by the Citrix Studio console. Configuration tasks include selecting the database (and obtaining the SQL scripting to initialize it), selecting the Citrix Configuration Service that holds the site configuration.

Note that some aspects of broker configuration (such as the port number on which the broker listens for SDK connections) cannot be configured with PowerShell cmdlets. These are configured through the use of the Broker Service executable. For more information, see about_Broker_PostInstallPreConfiguration.

A further important aspect of site management concerns the way in which machines providing resources identify the delivery controllers to which they belong. For more information, see about_Broker_ControllerDiscovery.

Managing XenDesktop sites involves the following SDK objects:

- BrokerSite
- BrokerController
- BrokerDBConnection
- BrokerDBSchema
- BrokerDBVersionChangeScript
- BrokerInstalledDbVersion
- BrokerServiceInstance
- BrokerServiceGroupMembership
- BrokerNameCache

For more information, see:

about_Broker_PostInstallPreConfiguration
about_Broker_ControllerDiscovery

Get-BrokerSite
Get-BrokerController
Get-BrokerDBConnection
Get-BrokerDBSchema
Get-BrokerDBVersionChangeScript
Get-BrokerInstalledDbVersion
Get-BrokerServiceInstance
Reset-BrokerServiceGroupMembership
Update-BrokerNameCache

about_Broker_ConfigurationSlots

March 11, 2024

Topic

Citrix Broker SDK - Configuration Slots and Machine Configurations

Short Description

Overview of assigning a collection of related settings to a desktop group.

Long Description

Collections of related settings may be applied to individual desktop groups through the creation of configuration slots and machine configurations.

A configuration slot defines a collection of related settings that are to be associated with that slot. Each machine configuration is associated with a single configuration slot and provides specific values for settings of that slot.

The SettingsGroup property of the configuration slot determines the particular collection of related settings that are associated with that slot. These groups are defined by Citrix and are not modifiable by administrators. For example, there is a particular group of Profile management specific settings that may be associated with a configuration slot. Because of the close association between a configuration slot and its

collection of related settings, the full set of configuration slots is created during the site creation.

Each machine configuration is associated with a single configuration slot. The machine configuration contains policy data that provides specific values for the settings associated with that configuration slot.

Every value set in a machine configuration's policy must belong to the configuration slot's settings group. Therefore the appropriate SDK snap-in must be used to create the policy data. For example, the Profile management snap-in must be used to create the policy data for a machine configuration associated with the Profile management configuration slot.

To have particular policy settings applied to the machines in a desktop group, a machine configuration is associated with that desktop group. A machine configuration may be associated with multiple desktop groups. A desktop group may be associated with multiple machine configurations.

When a machine configuration is associated with a desktop group, the configuration inherits the delegated administration restrictions of the desktop group. Thus, if a machine configuration is associated with multiple desktop groups, an administrator can only modify the policy data of the configuration if the administrator has permission to modify every one of those desktop groups.

For detailed information about defining and assigning machine configurations, see:

help New-BrokerMachineConfiguration
help Add-BrokerMachineConfiguration

See Also

- New-BrokerConfigurationSlot
- New-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- Import-BrokerDesktopPolicy

about_Broker_ControllerDiscovery

March 11, 2024

Topic

Citrix Broker - Configuring Controller Discovery

Short Description

Describes the way that machines providing published resources discover delivery controllers.

Long Description

In order for the broker to be able to connect users to desktops and applications, the machines from which they are published must register (that is, establish communication) with the broker on an appropriate delivery controller in the delivery site.

The default operation, whose configuration is described in this topic, is to use information from the registry. This is referred to as registry-based controller discovery. The registry information can be supplied when installing the delivery agent software on each machine or it can be supplied through group-policy.

If machines are provisioned using quick deploy, information about delivery controllers is stored in a special "identity disk"attached to the VM.

Finally, in some deployments, the use of an Organizational Unit (OU) in Active Directory (AD) may be preferred. This is referred to as AD-based controller discovery. In this case, you must configure the GUID of the OU in the machines' registries. Such configuration is not described in this topic.

To perform registry-based controller discovery, run the PowerShell script called Set-ADControllerDiscovery.ps1 that is installed on each controller in the folder:

$Env:ProgramFiles\Citrix\Broker\Service\Setup Scripts

For more information, run this script with the -help parameter.

about_Broker_Desktops

March 11, 2024

Topic

Citrix Broker SDK - Desktops

Short Description

Describes desktop concepts and usage.

Long Description

A desktop is a machine that is able to run a Microsoft Windows desktop environment (with a shell, icons and taskbar) or individual applications (seamlessly integrated with the local desktop). The configuration of the desktop determines whether it can run only desktop environments, only applications, or both desktops and applications. Machines running workstation operating systems are able to run one session at a time (single-session), whereas machines running server operating systems have the ability to run multiple simultaneous sessions (multi-session).

A key aspect of desktops is how they are assigned (or allocated) to users. Two allocation types are supported:

- Random/Shared - A user is assigned a desktop at random from a pool of shared desktops. Multi-session desktops are able to run sessions to multiple users simultaneously, whereas single-session desktops can only run one session at a time, and are returned to the pool when the user logs off. Single-session shared desktops usually discard user data stored on them after the user logs off. Multi-session shared desktops, however, do not tend to discard user data after a log-off, as this is only possible when the desktop is rebooted by a reboot schedule.
- Permanent/Private - A private desktop is permanently assigned to a specific user and data stored on it is retained across logons and restarts. A private desktop can have users assigned explicitly or on first use.

Desktop Groups

Desktops are collected together in desktop groups, and these provide a flexible grouping mechanism that can be used to associate:

- Desktops running on a particular type of machine
- Desktops with particular software installed
- Desktops for a set of users
- Desktops accessed in a similar way
- Desktops configured in a particular way
- Any combination of the above

Each desktop group can only contain one type of desktop, determined by its AllocationType and SessionSupport properties.

When assigning shared desktops or assign-on-first-use (AOFU) desktops to users, the set of candidates comes from available desktops in a particular desktop group.

You configure power management policy for single-session desktop groups, including peak and off-peak settings, for each desktop group. See about_Broker_PowerManagement for details.

Creation Of Desktops

Desktop objects are created automatically when a machine is added to a desktop group. The type of desktop is determined by the AllocationType property of the desktop group.

In order for a machine to be added from a catalog, the machine must be compatible with the desktop group. For this to be true, the catalog's AllocationType must be compatible, and the SessionSupport property must match.

Note: Because the session support and functional level of the machine are determined by the software on the machine (operating system and Citrix VDA, respectively), the SessionSupport and MinimalFunctionalLevel of the catalog and desktop group may match, but not be compatible with the machine. In this case any attempt of registration by the machine will fail.

You can add machines explicitly using the Add-BrokerMachine cmdlet, or a number of free machines can be acquired from a catalog using the Add-BrokerMachinesToDesktopGroup cmdlet. A machine can only be associated with one desktop group at a time, and has a DesktopUid property that references the corresponding desktop object. You can also associate desktops to machines with their SID properties.

Desktop objects are deleted when the machine is removed from the desktop group (using the Remove-BrokerMachine cmdlet). Desktops are also deleted

when the desktop group containing them is deleted (using the Remove-BrokerDesktopGroup cmdlet). A desktop cannot be removed from a catalog while it is in a desktop group.

Shared (Random) Desktops

Shared desktops are published to users using entitlement policy rules. Each entitlement rule allows access to a single session on a desktop machine, selected at random from the available desktops in a desktop group (with a preference for desktops that are powered on). If there are no available desktops, launching the session fails. See about_Broker_EntitlementPolicy and about_Broker_PowerManagement for details.

Private (Static) Desktops

You can assign private desktops to users explicitly or automatically, with the AOFU feature. It is also possible to assign private desktops to particular clients (through IP or client name).

You explicitly assign machines or desktops to users with the Add-BrokerUser cmdlet. Machines can be assigned to users before the machine has been added to the desktop group (desktop created), but otherwise the effect is the same.

With Add-BrokerUser you can assign a desktop to multiple users or user groups. If the desktop has single-session support then the desktop will be visible to multiple users, but only one user can log on to the desktop at any time.

Assignment policy rules allow you to use AOFU to assign desktops to users in a desktop group. When a user specified in an assignment policy rule launches a session, and if the user does not already have an assigned desktop, the broker selects an available desktop at random from the desktop group and permanently assigns it to that user. Once assigned in this way, the desktop behaves as though the assignment was made explicitly by an administrator. See about_Broker_AssignmentPolicy for details.

It is possible to assign more than one desktop to a user. You can achieve this by explicit assignment, multiple assignment policy rules, or the MaxDesktops property of an assignment policy rule.

Desktop Configuration

When presented to the user, desktops are identified by:

- An icon (IconUid property)
- A name (PublishedName property)
- A description (Description property)

When starting the session, you can configure two connection settings:

- The color depth used at the start of the session (ColorDepth property)
- Whether SecureICA encryption is required (SecureIcaRequired property)

Each desktop group provides default values for these settings, but you can override them if the desktop has more specific settings (PrivateDesktop), or use an entitlement policy rule with more specific settings (SharedDesktop).

AOFU desktops can inherit these settings from the assignment policy rule when the assignment to the user takes place.

Maintenance Mode

There are times when it is necessary to disable desktops. You can do this by setting the InMaintenanceMode property of a desktop to $true. This puts it into maintenance mode. The broker excludes single-session desktops in maintenance mode from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Desktops in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

Note that disabling desktop groups, entitlement policy rules, assignment policy rules, or applications are other ways of disabling aspects of brokering.

Desktop Status

Once desktops are created, you can query the configuration and state information for different kinds of desktop, or retrieve more information about desktops using the Get-BrokerMachine cmdlet. To get details of any sessions running on the desktops, use the Get-BrokerSession cmdlet.

You can also group desktops by a specific property, counting the number of desktops with each value using the Group-BrokerMachine cmdlet. This can provide useful summary statistics.

Desktop Usage

Every hour the broker records how many desktops from each desktop group are in use, and the Get-BrokerDesktopUsage cmdlet returns this information. Analyze historical usage records to understand desktop usage patterns and help with the choice of idle pool and buffer settings.

Desktop Conditions

CPU usage, ICA latency, and profile logon times of desktops are monitored. When one of these values exceeds a threshold (configured by policy), the condition is flagged in the DesktopConditions property of the desktop. When the value drops below the threshold again, the condition is cleared. Use Get-BrokerMachine or Group-BrokerMachines cmdlets to query this information.

See Also

- about_Broker_Concepts
- about_Broker_Applications
- about_Broker_EntitlementPolicy
- about_Broker_AssignmentPolicy
- Add-BrokerMachine
- Add-BrokerMachinesToDesktopGroup
- Remove-BrokerMachine
- Add-BrokerUser
- Set-BrokerMachine
- Set-BrokerPrivateDesktop
- Get-BrokerMachine
- Group-BrokerMachine
- Get-BrokerDesktopUsage

about_Broker_EntitlementPolicy

March 11, 2024

Topic

Citrix Broker SDK - Desktop and Application Entitlement Policy

Short Description

Controls end-user entitlement to desktop and application sessions provided from a pool of shared machines.

Long Description

The site's entitlement policy defines rules controlling users' entitlements to desktop and application sessions from pools of shared machines. Each pool is defined by a desktop group.

The entitlement policy comprises a set of rules. Each rule grants users a single entitlement to a desktop or application session in a specified desktop group. The policy can be applied only to groups of desktop kind Random. For desktop entitlements multiple rules can apply to the same group, however for application entitlements only a single rule can apply to a given group.

When the user starts a session by selecting an entitlement the behavior depends on the session-support property of the desktop group:

- For single-session groups the user is temporarily assigned a machine selected at random from the group to provide their session. When the session ends, the machine is returned to the pool of available machines.
- For multi-session groups the user session is provided by the machine that is least loaded within the group when the session is launched.

If multiple desktop entitlement rules for the same group contain the same user, the user can have as many desktop sessions from the group concurrently as they have entitlements.

Although only a single application entitlement rule can be defined for a group, a user can still launch multiple applications from that group because the applications all run within that entitlement's single session.

Rules for desktop and application session entitlements are distinct. Desktop entitlements are managed through the BrokerEntitlementPolicyRule SDK object, and application rules through the BrokerAppEntitlementPolicyRule object.

Desktop entitlement rules can be created only for desktop groups with delivery types DesktopsOnly or DesktopsAndApps, whereas an application entitlement rule can be created only for delivery types AppsOnly or DesktopsAndApps.

For desktop groups with delivery type DesktopsAndApps, typically one or more desktop session entitlement rules together with a single application session entitlement rule exist.

For an entitlement granted by the entitlement policy to be available to a user, the site's access policy must also grant them access to the desktop group.

Entitlement Policy Rules

Each entitlement policy rule has the following key properties:

- The desktop group to which it applies (one rule only ever applies to one group)
- The users to whom the entitlement is granted

Additionally for desktop entitlement rules, the following properties exist:

- The published name of the entitlement (visible to the user)
- Any properties that a desktop session launched using the entitlement should use that differ from the defaults specified on the desktop group

If multiple desktop entitlements are available to a user from the same group the resultant desktop session properties may differ depending on which entitlement the user selects to start the session.

Each rule can be individually enabled or disabled. A disabled rule is ignored when the entitlement policy is evaluated.

User Filters (Full)

Each rule has two user filters, an include filter and an exclude filter:

- The include filter contains users and user groups that are granted an entitlement to a session
- The exclude filter contains users and user groups that are denied an entitlement to a session

If the include filter of a rule contains multiple instances of a user (either explicit or implicit), they get only one entitlement by that rule.

Entries in the exclude filter take priority, so if a user appears explicitly or implicitly in both filters, access is denied. Typically, you use this filter to exclude a user or group of users who would otherwise gain access because they are members of a user group specified in the include filter.

Because all rules are independently evaluated, the exclude filter can only exclude users who would otherwise gain an entitlement through the same rule's include filter. That is, if a user is in a rule's include filter but not its exclude filter, the rule is guaranteed to grant that user a session entitlement irrespective of whether the user appears in the exclude filter of other rules.

If a filter contains a user group that contains other users and groups, the filter implicitly includes all of those users and groups.

By default the exclude filter is disabled.

To maintain entitlement policy rules that can be fully displayed and edited with Citrix Studio, use the simplified user filter model below and do not use the exclude filter.

User Filters (Simplified)

The included user filter described above also supports a simplified usage model where the filter itself is disabled. When this is done, any user who has access to the desktop group through the access policy is implicitly granted an entitlement to a session through the entitlement policy rule without the need to list the user in the rule's include filter.

This is useful in cases where the access policy for the desktop group already explicitly specifies the users who should have access.

Even if the include filter is disabled, the exclude filter can still be used to deny the entitlement from users who would otherwise gain access through the access policy alone.

Additional Desktop Entitlement Rule Properties

Desktop entitlement rules specify the following additional properties:

- PublishedName
- Description
- IconUid
- ColorDepth
- SecureIcaRequired

The published name, description, and icon UID properties apply to the desktop entitlement itself and determine the way in which the entitlement is presented to the user in, for example, StoreFront.

The color depth and secure ICA properties apply to the desktop session that is obtained when the entitlement is used.

In all cases, these properties can be explicitly specified. However, a null value (the default) means that the corresponding property is taken from the desktop group to which the rule applies. This inheritance from groups is dynamic; if the property of the group changes, the property of the entitlement changes too.

Notes

If a rule grants an entitlement to a user group, the session entitlement applies to the individual user who selects the entitlement. However, this does not prevent a different user in the same user group from using the same entitlement concurrently. So, a rule that grants an entitlement to a user group containing multiple users allows each user concurrent access to a single session from the desktop group.

The total number of entitlements defined by the policy may exceed the number of machines available, or the maximum allowed sessions, from the desktop group. A user attempting to use an entitlement when no further resources are available receives a no-desktop-available error.

If a session launched through an entitlement is active when the entitlement rule is deleted, the session continues unaffected. However:

- When the user ends the session, they cannot start a new one if the deleted rule was their only entitlement to a session in that group
- If the user disconnects the session, they cannot reconnect to it

See Also

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_Applications
- New-BrokerEntitlementPolicyRule
- Get-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule
- New-BrokerAppEntitlementPolicyRule
- Get-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule

about_Broker_ErrorHandling

March 11, 2024

Topic

Citrix Broker SDK - Error Handling

Short Description

Describes broker errors generated by cmdlets and how to access them.

Long Description

The broker SDK cmdlets report errors through the class SdkErrorRecord, which is a subclass of the standard Powershell error record class System.Management.Automation.ErrorRecord. SdkErrorRecord contains:

- A short string to describe the error status code. This is implemented as a public property named Status.

- A dictionary of key-value pairs containing additional data specific to the cmdlet. This is implemented as a public property named ErrorData of type Dictionary<string, string>.

The error status property always has a value. Populating the error data dictionary is optional. The number of entries within the dictionary, the content of the entries, and the exact format of the key and value data is specific to each cmdlet.

You can access an SdkErrorRecord object using the standard Powershell cmdlet ErrorVariable parameter. The type of object returned by ErrorVariable depends on whether the error is terminating or non-terminating.

Non-Terminating Errors

For non-terminating errors, each object in the returned ErrorVariable array is simply an instance of type SdkErrorRecord.

Terminating Errors

For terminating errors, the object returned by ErrorVariable is of type System.Management.Automation.CmdletInvocationException.

For non-terminating errors that are escalated as terminating errors (through the "ErrorAction stop"argument), the object returned by ErrorVariable is of type System.Management.Automation.ActionPreferenceStopException.

CmdletInvocationException and ActionPreferenceStopException are subclasses of the base class System.Management.Automation.RuntimeException, which exposes the SdkErrorRecord object through its ErrorData property.

Class SdkOperationException

SdkErrorRecord's Exception property holds an instance of custom exception class SdkOperationException, which also contains the error status code and data dictionary from SdkErrorRecord.

The SDK cmdlets generate errors in response to exceptions generated by the underlying system or by the cmdlet detecting errors locally and instantiating appropriate exception types. Such exceptions, which represent the original cause of the terminating error, are specified in SdkOperationException's InnerException property.

For terminating errors, use Powershell scripts to trap SdkOperationException and access additional error information and the originating exception.

Review Of Powershell Error Handling Behavior

Powershell scripts can access error information using the following methods:

- Read error records from the $error arraylist.
- Get the cmdlet to return error records using the standard ErrorVariable cmdlet parameter.
- Use trap blocks to catch exceptions for terminating errors.

The type of the error record that Powershell puts in the $error arraylist and returns through the ErrorVariable cmdlet parameter depends on whether the error is terminating or non-terminating, and whether the ‘‘ErrorAction continue’’or ‘‘ErrorAction stop’’cmdlet parameters are specified (that turn terminating errors into non-terminating ones, and vice versa).

For the combinations of error types (terminating, non-terminating) and error actions (stop, continue), the following statements describe the relationship between:

- The type of object contained in the Powershell $error arraylist.
- The type of object returned by the ErrorVariable cmdlet parameter (assuming the script can continue after a terminating error).
- The type of the exception that is thrown (where applicable).

Non-terminating errors with ErrorAction=Continue, $error type=SdkErrorRecord, and ErrorVariable type=SdkErrorRecord do not throw an exception.

Terminating errors with ErrorAction=Stop, $error type=ErrorRecord, and ErrorVariable type=CmdletInvocationException throw an SdkOperationException exception.

Non-terminating errors with ErrorAction=Stop, $error type=ErrorRecord, and ErrorVariable type=ActionPreferenceOperationException do not throw an exception.

Terminating errors with ErrorAction=Continue, $error type=ErrorRecord, and ErrorVariable type= CmdletInvocationException throw an SdkOperationException exception.

ActionPreferenceOperationException and CmdletInvocationException are subclasses of System.Management.Automation.RuntimeException.

Accessing Error Record Data

The Powershell code sample below demonstrates how to access error information programmatically with the ErrorVariable cmdlet parameter and a trap block.

```
# Trap exceptions generated from terminating errors
trap  [Exception] {

    write ""
    write "TRAP BLOCK : BEGIN"
    if($_.Exception.GetType().Name -eq "SdkOperationException")
    {

        $sdkOpEx = $_.Exception
        # show error status
        write $("SdkOperationException.Status = " + $sdkOpEx.Status)
        # show error data dictionary
        write $("SdkOperationException.ErrorData=")
        write $("SdkOperationException.InnerException = " + $sdkOpEx.
            InnerException)
        $_.Exception.ErrorData
     }

    continue #could also call break here to halt script execution
    write "TRAP BLOCK : END"
 }
```

```
#############################################################################

## Run tests 1 to 4, below, in turn to examine terminating and
## non-terminating error behavior.
#############################################################################
```

```
#
# Test 1: Invoke cmdlet that generates a terminating error:
#         New-BrokerCatalog throws terminating error if a
#         catalog with the supplied name already exists.
#
#New-BrokerCatalog -Name "AlreadyExists" -AllocationType Random -
    ProvisioningType Manual -SessionSupport SingleSession -
    PersistUserChanges OnLocal -MachinesArePhysical $true -ErrorVariable
     ev
```

```
#
# Test 2: Force script execution to continue after a terminating error.
#
#New-BrokerCatalog -Name "AlreadyExists" -AllocationType Random -
    ProvisioningType Manual -SessionSupport SingleSession -
    PersistUserChanges OnLocal -MachinesArePhysical $true -ErrorVariable
     ev -ErrorAction continue
```



```
#
# Test 3: Invoke cmdlet that generates a non-terminating error:
#         Get-BrokerCatalog generates a non-terminating error if a catalog
#         with the specified name doesn't exist.
#
#Get-BrokerCatalog -Name "IDontExist" -ErrorVariable ev
```

```
#
# Test 4: Force script execution to halt after a non-terminating error.
#
#Get-BrokerCatalog -Name "IDontExist" -ErrorVariable ev -ErrorAction "stop"
write ""
write "GET ERROR INFORAMTION: BEGIN"
$sdkErrRecord = $null
if($ev[0].GetType().Name -eq "SdkErrorRecord")
{

    $sdkErrRecord = $ev[0]
 }

elseif($ev[0].GetType().BaseType.FullName -eq "System.Management.Automation.RuntimeException")
{

    $sdkErrRecord = $ev[0].ErrorRecord
 }

else
{

  write ("UNKNOWN ERROR VARIABLE TYPE:")
  $ev[0].GetType().Name
 }

if($sdkErrRecord -ne $null)
{

    write ("Have sdkErrRecord:")
    write ("  Type Name = " + $sdkErrRecord.GetType().FullName)
    write ("  Status = " + $sdkErrRecord.Status)
    write ("  Exception type = " + $sdkErrRecord.Exception.GetType().FullName)
    write ("  ErrorData = ")
    $sdkErrRecord.ErrorData
    write ("  FullyQualifiedErrorId = " + $sdkErrRecord.FullyQualifiedErrorId)
 }

write "GET ERROR INFORAMTION: END"
```


about_Broker_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value
of the elements. You can specify a different sort order by qualifying
the name with an ordered list of elements or their numeric values,
or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and
supports combination of conditions with -and and -or, and grouping
with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use
script blocks in most cases. This can be easier to read as it reduces
quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double
quotes, and omit the quotes altogether for many strings. The order of
parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell
handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape
the wildcards using the normal PowerShell escape mechanisms (see
about_Escape_Characters), or switch to a filter expression and the -eq
operator:


```
# Escape examples
Get-<Noun> -Company "Abc[*]"                    # Matches Abc*
Get-<Noun> -Company "Abc`*"                     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}

Get-<Noun> -Filter { Shape -like ‘C*’}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
```



```
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{


```
break;
}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

about_Broker_GroupPolicy

March 11, 2024

Topic

Citrix Broker SDK - Group Policy for VDA Configurations

Short Description

Overview of Citrix Group Policy settings that control VDA behaviors.

Long Description

A VDA configuration policy is a collection of settings that specify how a VDA should initialize its environment when a user logs on. Settings that are to be enforced when a specific user session is created are user settings. Settings that are always applied to all user sessions are computer settings.

Multiple policies contianing multiple settings can be defined. Filters can be specified in a policy to define how the settings included in the policy are to be applied, based on data such as the user name, the name of the client, from which the user session is originated.

There is a policy processing engine (CSE) running on each VDA. The engine is responsible for processing the policies downloaded to the VDA from the site, to which the VDA is registered, and/or the domain the VDA is in. The engine uses policies from all the sources to calculate a net result, which is a single set of unique settings that can be used by the VDA to initialize the environment when a user logs on.

A policy object contains a set of properties that describe the object. It also contains a set of settings, which are specific VDA configuration settings, as well as a set of filters, which specify how the settings in the policy are applied.

- A setting is a single piece of data that specifies a certain behavior for a user session. For example, a wallpaper setting is used to control if a wallpaper should be displayed when a user logs on.
- A filter is a statement that specifies how the settings in the policy are to be applied. When evaluted with the input data required by the filter, the statement yields a Boolean value. A true result tells the policy engine that the settings should be applied. A false result tells the policy engine to not apply the settings.

To define a VDA configuration policy, specify a name for the policy and a priority that indicates a precendence for the settings in the policy when the policy is evaluated togehter with other policies by the Citrix policy engine (CSE) running on a VDA. A name and an optional brief description can be specified to help identifying the policy and its purpose.

Four types of objects are defined to store policy data. They are policy sets, policies, settings, and filters.

A policy set is a container of a collection of policies. For each site, there is always a default policy set defined and this policy set contains all the site policies.

A policy is a collection of settings and filters.

A setting specifies a configuration that will be honored and enforced by a VDA component.

A filter is a predicate that specifies the condition for the settings in the policy to be applied or not applied.

There can be multiple policies in a policy set. There can be multiple settings in a policy. There can be multiple filters in a policy. A setting can be used in multiple policies but can only be used once in a policy.

Multiple filters can be specified in a policy and the final result of combining all the filters decides if the policy should be applied or not.

For detailed information about defining policy objects, see:

help New-BrokerGpoPolicySet
help New-BrokerGpoPolicy
help New-BrokerGpoSetting
help New-BrokerGpoFilter

See Also

- New-BrokerGpoPolicySet
- New-BrokerGpoPolicy
- New-BrokerGpoSetting
- New-BrokerGpoFilter
- Get-BrokerGpoPolicySet
- Get-BrokerGpoPolicy
- Get-BrokerGpoSetting
- Get-BrokerGpoFilter
- Set-BrokerGpoPolicy
- Set-BrokerGpoSetting
- Set-BrokerGpoFilter
- Remove-BrokerGpoPolicySet
- Remove-BrokerGpoPolicy
- Remove-BrokerGpoSetting
- Remove-BrokerGpoFilter

about_Broker_Licensing

March 11, 2024

Topic

Citrix Broker - Licensing

Short Description

Overview of broker licensing configuration.

Long Description

As part of the licensing setup for a site, the types of product licenses used by the broker when creating connections to virtual desktops or applications can be configured using the Central Configuration Service SDK.

The following licensing related properties can be specified using the Set-ConfigSite cmdlet:

- LicensingModel - Sets the licensing model to use. Values can be 'Concurrent' or 'UserDevice'.
- ProductCode - Specifies which product license is supported by the site. Values can be "MPS" for XenApp licenses or "XDT" for XenDesktop licenses.
- ProductEdition - Sets the licensing edition to use.

A license matching the specified model, product code, and edition must be available within the site's license server in order for the broker to grant licenses.

These properties are part of the site object returned by the Get-ConfigSite cmdlet.

Concurrent License Model

A concurrent license is tied to a XenDesktop session. When a user launches a session, a license is checked out to that session. When a user logs off from a session, the license is checked back in again, making it available for another session.

User Device License Model

With user device licensing, the license server automatically assigns licenses to users or devices based on usage:

- User licensing allows users to access their desktops and applications from multiple devices.
- Device licensing allows multiple users to access their desktops and applications from a single device.

When users or devices connect to an application or desktop, they consume a license for a 90-day license assignment period. The assignment period begins

when a connection is made, is renewed to the full 90 days during the life of the connection, and expires (allowing reassignment) 90 days after the last connection terminates. A license assignment can be manually ended before the 90-day period elapses using the udadmin command line installed on the license server.

Licensing State

The broker site contains the following properties related to licensing state:

- LicensingGracePeriodActive - Reports if the broker is in licensing grace period.
- LicensingOutOfBoxGracePeriodActive - Reports if the broker is in out-of-box grace period.
- LicensingGraceHoursLeft - The number of grace hours remaining, if the broker is in grace period, else it is null.
- LicensedSessionsActive - The number of active, licensed sessions.
- LicenseGraceSessionsRemaining - The number of concurrent grace sessions available for the site Product Code, if the broker is in licensing grace period, else it is null.

These properties are part of the site object returned by the Get-BrokerSite cmdlet.

License Server Test

The broker SDK cmdlet Test-BrokerLicenseServer checks whether or not a given server can be used as a license server by the broker.

Resetting License Server Connection

The broker SDK cmdlet Reset-BrokerLicensingConnection resets the broker's connection to the license server.

License Burn-In Date

The version of the product that is supported within the site is denoted by a licensing burn-in date. This date can be accessed through the

LicensingBurnInDate field of the site object returned by the Get-ConfigSite cmdlet.

See Also

- Test-BrokerLicenseServer
- Reset-BrokerLicensingConnection
- Get-BrokerController
- Set-ConfigSite
- Get-ConfigSite
- Get-BrokerSite

about_Broker_Machines

March 11, 2024

Topic

Citrix Broker SDK - Machine Object

Short Description

Describes machine concepts and usage.

Long Description

A machine represents a physical or virtual machine that can be used to provide a user with one or more desktops, applications or both. When a machine is created, you must assign it to a catalog, which defines how the machine is allocated to a user (static or random), the session support it provides (single-session or multi-session), how the machine's disk images are created and managed (PVS, MCS or manually) and the expected functional level. If the machine is virtual but not provisioned by MCS, you must also assign it to a hypervisor connection, which represents the hypervisor (or pool of linked hypervisors) that runs the virtual machine.

Creating a machine object is the first step in the broker SDK towards configuring a physical or virtual machine to provide desktops and/or applications to users. A machine must be added to a desktop group before it is able to be used(see about_Broker_Desktops). To add machines to a desktop group, the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets can be used. This creates a desktop object corresponding to the machine.

Catalogs

When a machine is created, you must assign it to a catalog. The catalog defines the behavior of the machine within a site as well as the expected functionality and properties of the machine:

- Allocation type: The catalog determines how the machines are allocated to the user. Allocation can be static or random. Static allocation is where the machine is permanently assigned to a specific user. Data stored is retained across logons and restarts.

  The other type of allocation is random, where a random machine is assigned to a user from a pool when a session is requested. The machine returns to the pool when the user logs off.

- How the machine is created: The catalog collects together machines that are created in the same way: either with PVS, MCS or manually.

- Physical or Virtual: A machine that is virtual can have its power state controlled and monitored by the system. Virtual machines must be associated with a hypervisor connection, either directly or, in the case of MCS provisioned machines, indirectly through the provisioning scheme. Single session virtual machines can be managed using power policy to automatically be turned on or off as needed. Machines marked as physical are not monitored or controlled as to their power state.

- How the users settings are stored: The catalog also determines how the users settings are stored, either on a Citrix Personal vDisk, on the machine□s local drive, or if the user settings are discarded.

- RemotePC: If the catalog is set up as a remote PC catalog, machines are added automatically upon registration based on the site configuration. In order for a catalog to be specified as a RemotePC catalog, the session support must be single session and the catalog must be set up for physical machines.

- Functional level: The functional level of a machine is determined by the version of the Citrix VDA software it is running. Some features are not supported in machines with lower functional levels. Catalogs can supply a minimal functional level, meaning any machines in the catalog with a lower functional level will be unable to register with the site.
- Session support: This can either be single-session or multi-session. Single-session machines can have an active session with up to one user at any time, whereas multi-session machines have the capability to have active sessions with multiple users simultaneously. The session support of a machine is determined by the variant of the VDA software component installed on the machine (either with single-session support or multi-session support). The multi-session VDA software may only be installed on server operating systems. The catalog session support must match the session support of the software installed on the machine for the machine to successfully register with the site.

Machine Status

After machines are created, you can query the configuration and state information using the Get-BrokerMachine cmdlet. The information the cmdlet can provide includes, but is not limited to, the following:

- Personal vDisk interactions and lifecycle: The current state of the personal vDisk can be obtained, as well as the configuration options of how the user data is persisted.
- Session properties: The properties of the current session for single-session machines can be obtained, such as ClientName and ClientAddress. To access session information on multi-session machines, the Get-BrokerSession cmdlet can be used.
- Application status: If the machine is configured to run applications, information can be found about the published applications running on the machine.
- Connection information: Information about the time and user of the last connection to the machine can be found, as well as information about the last deregistration.

For an exhaustive list of the properties of a machine that can be queried, see the Get-BrokerMachine cmdlet.

Machine Configuration

Machine settings can be changed and configured once the machine object has been created, as long as the changes are compatible with the catalog the machine is in. For example, more users can be assigned to the machine than were initially assigned when creating the machine object, this is done with the Add-BrokerUser cmdlet.

For a full list of the machine configuration options available, see the Set-BrokerMachine command.

Maintenance Mode

There are times when it is necessary to disable machines. This can be achieved by setting the InMaintenanceMode property to $true. This puts the machine into maintenance mode. With single-session machines, this means that the broker excludes the machine from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Machines in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

See Also

- about_Broker_PowerManagement
- about_Broker_Desktops
- New-BrokerMachine
- Add-BrokerMachine
- Add-BrokerMachinesToDesktopGroup
- Remove-BrokerMachine
- Get-BrokerMachine
- Set-BrokerMachine

about_Broker_Policies

March 11, 2024

Topic

Citrix Broker SDK - Access, Entitlement, and Assignment Policies

Short Description

Overview of the site policies that control users' access to desktop and application sessions.

Long Description

For an end user to access a desktop or application resource within a site, they must have both an entitlement to use the resource, and have access to the desktop group that contains the resource.

Entitlements to use resources can be granted by one of the following means:

- The site entitlement policy grants entitlements to launch a shared desktop or application session from a pool of shared machines.
- The site assignment policy grants entitlements for "self service" permanent assignment of machines to users for running desktop or application sessions, and is referred to as "Assign On First Use"(AOFU)
- Machines can be permanently assigned ("pre-assigned") to users by the administrator to run either desktop or application sessions.
- Machines can be configured to allow automatic permanent assignment to their normal user (using the RemotePC feature).

A user must also be granted access to the desktop group that contains the resource. These access rights are controlled by the site's access policy.

The access policy controls access using details of the user's device such as whether it's connected over a local area network (LAN) or connected through Access Gateway, the user device's name, IP address or subnet, and the requested connection protocol. The user's identity can also feed into the access check allowing, for example, certain users access to resources only when locally connected to the site, but others full remote access.

Access and entitlements can be combined to allow rich and fine-grained control over which users have access to site resource from any given user device or location.

Each site has a single access policy, entitlement policy, and assignment policy. Each policy comprises a set of rules. Policies are defined by adding, removing, or changing rules.

Each site policy can also be viewed as a set of distinct policies each relating to a single desktop group. In general a group has one or more policy rules that relate to it, however each rule relates to only a single group. Thus the rules that grant entitlement and access rights to a desktop group define the policy for that group and that group only; changing this policy has no impact on the entitlement and access rights for any other other group in the site.

For detailed information about defining policy rules, see:

help New-BrokerAccessPolicyRule
help New-BrokerEntitlementPolicyRule
help New-BrokerAssignmentPolicyRule
help New-BrokerAppEntitlementPolicyRule
help New-BrokerAppAssignmentPolicyRule

The mapping of policies to the resources that they make available within a site is described briefly below. For specific information on configuring each category of resource, consult the more detailed help topics listed.

Shared Desktop And Application Sessions

To grant access to a group of shared machines, use the access and entitlement policies:

- The access policy grants access to the desktop group containing the machines to be shared.
- The entitlement policy grants an entitlement to use one or more machines in the group to specified users or groups of users.

Groups of shared machines can be used to deliver full desktop or seamless application sessions, or both.

For more detailed information about configuring shared machines, see:

help about_Broker_AccessPolicy
help about_Broker_EntitlementPolicy

Pre-Assigned Private Machines

To grant access to private machines, use the access policy and a machine assignment:

- The access policy grants access to the desktop group containing the machines.
- The assignment links the desktop to a specified user. You can assign a machine to just one user, multiple users or user groups. However, for single-session machines, only one user can access the machine at a time.

Private machines can be used to deliver full desktop or seamless application sessions (but not both).

For more detailed information about configuring private machines, see:

help about_Broker_AccessPolicy
help Add-BrokerUser

Assign-On-First-Use (Aofu) Machines

To grant access to a desktop group containing assignable machines, use the access policy and the assignment policy:

- The access policy grants access to the desktop group containing the pool of machines.
- The assignment policy grants users a self-service entitlement to pick one or more machines from the pool.

AOFU machines can be used to deliver full desktop or seamless application sessions (but not both from the same desktop group).

For more detailed information about configuring AOFU desktops, see:

help about_Broker_AccessPolicy
help about_Broker_AssignmentPolicy

Remote Pc Machines

The RemotePC feature allows existing physical machines to be assigned automatically to their normal user thus allowing them remote access to their own machine but without the need for the administrator to individually configure access to each machine.

For more detailed information about configuring the Remote PC feature, see:

help about_Broker_RemotePC

See Also

- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_AssignmentPolicy
- about_Broker_RemotePC
- New-BrokerAccessPolicyRule
- New-BrokerEntitlementPolicyRule
- New-BrokerAssignmentPolicyRule
- New-BrokerAppEntitlementPolicyRule
- New-BrokerAppAssignmentPolicyRule
- Add-BrokerUser

about_Broker_PostInstallPreConfiguration

March 11, 2024

Topic

Citrix Broker SDK - Post-Installation Configuration

Short Description

Describes how to configure the Citrix Broker Service port numbers, URL reservations, and Windows Firewall exclusions.

Long Description

The XenDesktop installer configures the Citrix Broker Service with information specified during the installation. To change that configuration, use the BrokerService.exe command-line tool on each controller you want to change.

The default installation location of BrokerService.exe is:

%ProgramFiles%\Citrix\Broker\Service\BrokerService.exe

BrokerService.exe supports the following optional command-line parameters:

-SdkPort <port> (default 80)

Configures the port on which the broker listens for requests from SDK cmdlets. If you change this default value, specify the new value in the AdminAddress parameter on broker cmdlets. For example, if you changed the port to 8080, specify it as follows:

Get-BrokerSite -AdminAddress localhost:8080

-VdaPort <port> (default 80)

Configures the port on which the broker listens for registration requests from broker machines.

-WiPort <port> (default 80)

Configures the port on which the broker listens for XML requests from StoreFront/Web Interface.

-WiSslPort <port> (default 443)

Configures the port on which the broker listens for SSL (Secure Socket Layer) XML requests from StoreFront/Web Interface.

-ConfigureFirewall

Configures Windows Firewall exclusions for the specified ports.

-Show

Shows the current configuration settings.

-Uninstall

Removes configuration settings, including Windows Firewall exclusions and URL reservations.

-LogFile <fileName>

Configures the file location for logging to a text file. The directory containing the log file must grant write access to the NetworkService account.

-Upgrade

Performs configurations required after an upgrade.

-Quiet

Suppresses console output for status messages.

-? or -Help

Shows usage information for the command.

about_Broker_PowerManagement

March 11, 2024

Topic

Citrix Broker SDK - Machine Power Management

Short Description

Describes power management of machines used for desktops and applications.

Long Description

The Citrix Broker Service is in day-to-day control of the power state of the configured desktop and application machines. The Broker Service can control several hypervisors, each hypervisor connection being handled by its own site service, so all Broker Service communication to the hypervisor is through one of the controllers in the site.

Hypervisor Connections

Each hypervisor, or pool of linked hypervisors, is described and configured through the XdHyp pseudo-drive and associated Hyp PowerShell commands (cmdlets) which are provided by the host service snap-in. When you have first created the hypervisor connection using the host service cmdlets, you can create a broker equivalent object that references the Hyp instance using a GUID value. Use the broker's HypervisorConnection object to nominate a preferred controller for direct communication with the hypervisor on behalf of all other controllers for day-to-day power actions and status requests.

Power Actions And Throttling

The site, through the Broker Service, can control the power state of the machines used by the site for desktops and applications. Power state changes can have a number of causes:

- Power policy rules, such as requests to shut down or suspend machines when user sessions on those machines end or are disconnected
- If allowed, user-driven desktop restarts
- Session launch requests requiring machines to be started
- Pool size management, which controls the number of running machines
- Direct administrator request using the SDK or Citrix Studio
- Reboot schedules and cycles
- Performing personal vDisk inventory activities
- Cleaning machines back to the golden master image state after they have been used

The power state changes of machines hosting desktops and applications are controlled using a queuing mechanism. Actions to change the power state are assigned a priority and are sent to the hypervisor according to a throttling mechanism. This avoids overloading the hypervisor.

The queuing and throttling mechanisms take place on a per-hypervisor-connection basis; each hypervisor connection's queue is dealt with independently. You can view the contents of the queues using the Get-BrokerHostingPowerAction cmdlet. This includes recently completed, in-progress, and pending actions (that is, those due to be sent to the hypervisor based on the throttling settings).

Each power action object comprises:

- The machine to be acted on (Name, DNS Name, Hosting Name)
- The action to be performed (TurnOn, TurnOff, Shutdown, Reset, Restart, Suspend, or Resume)
- The action's priority (Base, the original priority, and Actual, the current priority)
- The action's state (Pending, Started, Completed, Failed, Canceled, Deleted, or Lost)
- Time stamps of the action's lifecycle points (when it was created, started, or completed)
- Any reason the action failed

The throttling of power actions is controlled by three metadata values on

the Hyp hypervisor connection object when accessed through the XdHyp pseudo-drive. The four values throttle power actions according to:

- The maximum absolute number of in-progress power actions
- The maximum number of in-progress power actions expressed as a percentage of the total number of machines controlled by the hypervisor connection
- The maximum number of new power actions sent to the hypervisor per minute
- The maximum number of in-progress PvD inventory activities expressed as a percentage of the total number of machines controlled by the hypervisor connection

You add power actions to the queue using the SDK's New-BrokerHostingPowerAction cmdlet. You cancel power actions in the queue using the Remove-BrokerHostingPowerAction cmdlet. You boost or reduce their priority using the Set-BrokerHostingPowerAction cmdlet.

You can also schedule power actions to be executed in the future, using the New-BrokerDelayedHostingPowerAction command. Only Shutdown and Suspend actions can be scheduled in this way. You can view these delayed power actions using Get-BrokerDelayedHostingPowerAction and cancel them with Remove-BrokerDelayedHostingPowerAction. When a delayed power action is ready to be executed it is deleted, and a corresponding normal power action is placed in the queue described above.

Power Policy

Policy rules associated with a desktop group allow you to change power states at configurable times after session state changes, typically a set number of minutes after session disconnection or session logout.

Note that these policy rules are defined directly as properties of the desktop group.

These policy actions allow the following operations to be specified:

- A power action to be performed at a defined period after a session is disconnected
- A power action to be performed at a defined extended period after a session is disconnected
- A power action to be performed at a defined period after a session is logged off

The two disconnect policy actions are designed to allow multi-stage policies such as initially suspending a machine shortly after a session disconnect occurs, and then later powering-off the machine if the session has not been reconnected.

At the set time after the session state change, the required action is added to the power action queue, and this is then throttled and processed as normal.

Pool Size Management

You can manage flexibly the number of machines running desktops and applications using the pool size. For any given hour of the day and day of the week, this is an absolute number of machines or the percentage of the total number of machines in the desktop group. The pool size specifies the total number of machines that are always running, regardless of whether they are in use or idle. (Note: The number of machines does not depend on their idle status, but this does affect the buffer size value, which is also used to manage pool sizes.)

To start or shut down desktop machines to achieve the desired pool size, the system places power actions in the queue. Standard throttling queue management sends these to the hypervisors. A single desktop group (and its pool) can span multiple hypervisors, so actions to start and shut down machines can be added to multiple queues.

Power Time Schemes

Each single-session desktop group can be associated with one or more power time schemes, each scheme covering a number of days of the week. The time schemes specify, for each hour of the day, whether that hour is peak or off-peak. They also specify the number of running unassigned machines maintained by the broker.

You can configure other settings, such as buffer size and any power policy rules differently for peak and off-peak hours. You can define the number of running machines, idle or in use, either as an absolute value or as a percentage of the desktop group size. Machines running desktops and applications are started (or shut down when not in use) to match the required pool size.

Each power time scheme comprises:

- The name of the scheme
- The pattern of days of the week covered by the scheme
- The set of hours considered peak and off peak
- The set of pool size values (one for each hour of the day)

The hours of the day used by time schemes are the hours in the time zone for the desktop group the scheme is associated with. You cannot associate one desktop group with multiple time schemes covering the same day of the week.

Buffer Size

In addition to the pool size, you can optionally configure two buffer sizes for each desktop group, one for peak hours and one for off-peak hours. The buffer size defines the minimum number of idle unassigned machines maintained by the broker and is specified as a percentage of the total machines in the group. These are running machines that are not used by any user session. The buffer size on its own never causes machines to shut down. It causes them to start up so a minimum number of idle machine is always available. The buffer size in conjunction with the pool size can cause machines running desktops or applications to be shut down.

Power Management Of Assigned Machines

Automatic power management for private desktop groups provides the ability to power on all assigned machines at the transition to a peak period and respectively power off all machines at the transition to an off-peak period.

If a machine is shut down during peak hours it will not be automatically powered on again, unless the AutomaticPowerOnForAssignedDuringPeak property on the desktop group is also enabled.

Note that all power management facilities apply only to single session machines.

Reboot Schedules

Reboot schedules are commonly used after image updates or to perform regular reboots of all machines in a desktop group or catalog to clear down problems resulting from a corrupt state or hung/faulty applications.

Reboot schedules allow distributing the reboot operation of all machines over a provided duration. Individual machine reboots are scheduled in a way

that attempts to maintain maximum availability of machines in the group as the reboots occur, and avoid boot storms that overload the underlying infrastructure.

Reboot schedules are the only form of automatic power management that can shut down a machine while users are logged on; however the administrator can provide a warning message to be displayed to end users at a specified period prior to the shutdown taking effect.

Reboot Cycles

Reboot cycles describe the dynamic execution of desktop group or catalog reboot operations. Reboot cycles can be created due to reboot schedules, or by on-demand reboot operations requested via the SDK.

RebootCycle objects encapsulate the details of the associated reboot operation and can be queried to show the current status.

Status

You can view the status of the hypervisor connection on the broker hypervisor connection object. You can obtain any hypervisor alerts using the Get-BrokerHypervisorAlert cmdlet. You can check the power state of machines running desktops or applications using the relevant Machine or Desktop objects.

See Also

- about_Broker_Concepts
- about_Broker_Machines
- about_HypHostSnapin
- New-BrokerHypervisorConnection
- Get-BrokerHypervisorConnection
- Set-BrokerHypervisorConnection
- Remove-BrokerHypervisorConnection
- New-BrokerHostingPowerAction
- Get-BrokerHostingPowerAction
- Set-BrokerHostingPowerAction
- Remove-BrokerHostingPowerAction
- New-BrokerDelayedHostingPowerAction

- Get-BrokerDelayedHostingPowerAction
- Remove-BrokerDelayedHostingPowerAction
- New-BrokerPowerTimeScheme
- Get-BrokerPowerTimeScheme
- Set-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme
- Remove-BrokerPowerTimeScheme
- Get-BrokerRebootCycle
- Set-BrokerRebootCycleMetadata
- Start-BrokerRebootCycle
- Stop-BrokerRebootCycle
- Get-BrokerRebootSchedule
- Set-BrokerRebootSchedule
- New-BrokerRebootSchedule
- Remove-BrokerRebootSchedule
- New-BrokerDesktopGroup
- Get-BrokerDesktopGroup
- Set-BrokerDesktopGroup
- Add-HypMetadata
- Remove-HypMetadata

about_Broker_RemotePC

March 11, 2024

Topic

Citrix Broker SDK - RemotePC

Short Description

Overview of the Remote PC feature.

Long Description

Remote PC allows automatic publishing of a user's physical desktop within a XenDesktop site, so that it can be accessed remotely. The configuration of

Remote PC within the Citrix Broker service specifies the rules and relationships that allow the machine to successfully register with a controller in the site, and be made available for the user to start remote sessions.

When Remote PC is configured, there are two steps required for publishing a machine as a Remote PC desktop.

First a VDA must be installed on the machine and it must be configured such that it registers with a controller in a site.

Second, a user must log on to the machine. When the Citrix Broker service detects an active console session for a user, it assigns the user to the machine and publishes it in a Remote PC desktop group.

Remote PC configuration consists of defining relationships between:

- Machines and catalogs
- Catalogs and desktop groups
- Desktop groups and assignment policy rules
- Assignment policy rules and users

The Citrix Broker service automates the assignment of Remote PC machines to users in two stages. The first stage automatically imports matching unconfigured machines into the site:

- Suitable machines are automatically added to a Remote PC catalog.
- The machines are temporarily configured to be in one of the Remote PC desktop groups associated with the catalog.

When a suitable user logs on to the console of the machine, the second stage of the automatic configuration is performed:

- The machine is configured to be in the desktop group that is appropriate for the user.
- The machine is assigned to the user that has logged on.
- The machine desktop is made available to the user remotely, configured to appear as the machine hostname.

Catalogs and desktop groups can be marked as participating in RemotePC automation with the 'IsRemotePC'property, but this property can only be set to true if various other properties of the catalog or desktop group are appropriate. Catalogs must be single-sessioned and contain physical machines. Desktop Groups must be single session, configured to deliver private assigned machines, delivering desktop sessions only.

Catalogs and desktop groups can have the 'IsRemotePC'property cleared to remove them from the RemotePC automation, but only when all RemotePC associations relating to them through RemotePCAccounts and catalog/group relationships have first been removed.

Machines And Catalogs

Mappings are defined between machines and Remote PC catalogs through the RemotePCAccount cmdlets.

The machine to catalog mappings support the automated addition of machines to catalogs.

Properties

RemotePCAccounts expose sets of included and excluded machine name filters specified in DOMAIN\MACHINE format. A MachinesIncluded or MachinesExcluded entry can include asterisk wildcards to generalize matches.

Each RemotePCAccount can specify the Distinguished Name (DN) of an AD container in addition to the machine name filters, in the RemotePCAccount Organisational Unit (OU) property, and this limits the machines that the account objects act on to those that reside at or below that container in the AD domain hierarchy. An AllowSubfolderMatches setting on the RemotePCAccount indicates whether the computer must exist directly within the container to trigger a match, or whether it can be in a child below the defined container in the AD hierarchy.

Note that the AD container component is optional and a special value of 'any'can be supplied in the OU field to permit the RemotePCAccount to automatically match regardless of the AD machine object location in the AD domain hierarchy. A match is still subject to machine name filtering.

The last component of the RemotePCAccount is the CatalogUid. This indicates which catalog the Remote PC automation should move the machine into when a match is found.

Constraints

The IsRemotePC setting must be enabled on catalogs before they can be specified in a RemotePCAccount.

There can be any number of RemotePCAccounts configured in the site as long as each specifies a unique OU. There can be only one RemotePCAccount with the 'any'OU.

Automation

When a machine matching the criteria set up in a RemotePCAccount instance registers with one of the brokers in the site, it is automatically added to the catalog defined by the RemotePCAccount. This can take up to 30 seconds to happen after the machine registers.

When the machine registration occurs, the machine details can match more than one RemotePCAccount instance, but the machine can only be placed into one catalog, so one RemotePCAccount instance is chosen from the list that best matches the machine. This choice is made according to the length in nodes of the DN of the AD container specification associated with the RemotePCAccount, so more specific child OUs override specifications for their parent OUs if both are present. The RemotePCAccount for the 'any'OU is always last and used only if no other instances match.

A Windows eventlog message is generated when an automated catalog assignment is performed.

Notes

The AD distinguished name for the container is checked when the RemotePCAccount is created, but if the container is subsequently moved or deleted, the site does not automatically accommodate this, and the RemotePCAccount must be changed or removed manually.

Related Cmdlets

- New-BrokerRemotePCAccount
- Get-BrokerRemotePCAccount
- Set-BrokerRemotePCAccount
- Remove-BrokerRemotePCAccount
- New-BrokerCatalog [-IsRemotePC <Boolean>]
- Set-BrokerCatalog [-IsRemotePC <Boolean>]

Catalogs And Desktop Groups

A Remote PC catalog may be associated with one or more Remote PC desktop groups. The catalog to desktop group associations support automated publishing of machines to users.

Usage

An association is formed via:

Add-BrokerDesktopGroup -RemotePCCatalog <Catalog> [-Priority <Int32>]

An association is broken via:

Remove-BrokerDesktopGroup -RemotePCCatalog <Catalog>

To find associated desktop groups:

Get-BrokerDesktopGroup -RemotePCCatalogUid <Int32>

Automation

When a machine in a Remote PC catalog is not already assigned to a user, Remote PC automation temporarily places the machine in one of the desktop groups associated with the catalog. The temporary placing of the machine in a group will be adjusted as needed when the machine assignment to a user is first made.

The desktop group chosen as the temporary home for a machine is according to the priority value supplied when making the association between the group and the catalog. The group with the highest priority (lowest numerical priority value) is chosen.

A Windows eventlog message is generated when an automated machine assignment is performed.

Priority

Each desktop group to catalog association has a priority value that can be specified when the association is made or defaults to lower than the lowest existing association priority (highest numerical value) or zero if no other association exists yet. The lower the priority numerical value, the higher the priority level. The priority value for the association is

used to choose the temporary desktop group to place a new RemotePC machine in when it first registers. It is also used to decide which group to finally place the machine in when the user that the machine is being assigned to is appropriate for more than one of the associated desktop groups, usually because the desktop groups are using AD security groups for the user associations with the desktop groups.

The priority values for each association between a desktop group and a catalog are automatically maintained as unique for each catalog, and priorities can be adjusted up or down accordingly by the system. The last association created without a specified priority is always arranged to have the least priority (the highest numerical priority value).

Notes

The temporary placement of the machines in a desktop group is not re-evaluated if settings change after the automatic placement.

Related Cmdlets

- Add-BrokerDesktopGroup -RemotePCCatalog <Catalog>
- Remove-BrokerDesktopGroup [-RemotePCCatalog <Catalog>]
- Get-BrokerDesktopGroup [-RemotePCCatalogUid <Int32>]
- New-BrokerCatalog [-IsRemotePC <Boolean>]
- Set-BrokerCatalog [-IsRemotePC <Boolean>]
- New-BrokerDesktopGroup [-IsRemotePC <Boolean>]
- Set-BrokerDesktopGroup [-IsRemotePC <Boolean>]

Desktop Groups, Assignment Policy Rules, And Users

Assignment policy rules are used to define sets of users allowed to be assigned to machines by Remote PC automation, and to determine which desktop group the machine is placed in when the user assignment is made.

Automation

A machine is automatically assigned to a user when the Citrix Broker service sees that the user has logged on to a RemotePC machine, and the user is amongst those configured in the desktop group assignment policy

rule. If this is the first assignment of a user to a machine, all the assignment policy rules for all groups associated with the machine are checked, and the machine can be moved to a different, more appropriate desktop group if needed.

A Windows eventlog message is generated when an automated assignment of a machine to a user is made.

Multiple Assignments

By default, multiple automatic assignments of users to the same machine can be established if multiple users log on to the RemotePC machine, but this can be disabled if desired using a registry setting (see CTX137805).

Notes

User to machine assignments can still be made and removed manually through the Powershell SDK for machines in Remote PC catalogs and desktop groups.

The automatic placement of a machine in an appropriate desktop group happens when the first automatic assignment of a user to the machine is made, and this placement will not be automatically updated if the configuration subsequently changes.

Only a single assignment policy rule is allowed for each RemotePC desktop group.

Related Cmdlets

- New-BrokerUser
- New-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule

Example

The following example creates a simple configuration that allows any user and machine in the current AD domain to participate in RemotePC.

```
# Create a Remote PC catalog
$catalog = New-BrokerCatalog -IsRemotePC $true
                             -SessionSupport SingleSession
                             -MachinesArePhysical $true
```



```
                                -AllocationType Static
                                -PersistUserChanges OnLocal
                                -ProvisioningType Manual
                                -Name RemotePCCatalog
```

```
# Create a Remote PC desktop group
$dg = New-BrokerDesktopGroup -IsRemotePC $true
                             -SessionSupport SingleSession
                             -DeliveryType DesktopsOnly
                             -DesktopKind Private
                             -Name RemotePCDesktopGroup
```

```
# Create an assignment policy rule for that desktop group allowing any
# domain user to match.
New-BrokerAssignmentPolicyRule -DesktopGroupUid $dg.Uid
                               -IncludedUsers 'domain users'
                               -Name RemotePCAPR
```

```
# Create a RemotePCAccount matching any unconfigured machine, causing
   the
# machines to be added to the catalog by Remote PC automation.
New-BrokerRemotePCAccount -OU 'any'
                          -CatalogUid $catalog.Uid
```

```
# Associate the desktop group and catalog to permit domain users to be
# automatically assigned to machines in that catalog
Add-BrokerDesktopGroup $dg -RemotePCCatalog $catalog
```

```
# Create an access policy rule, allowing access to the users to the
# Remote PC desktop group.
New-BrokerAccessPolicyRule -IncludedUsers 'domain users'
                           -DesktopGroupUid $dg.Uid
                           -IncludedUserFilterEnabled $true
                           -Name RemotePCAccessPolicyRule
```

about_Broker_ServiceConfigurationData

March 11, 2024

Topic

Broker Service Configuration Data

Description

The detailed behavior of the Broker service can be configured using settings available via the Set/Get-BrokerServiceConfigurationData SDK cmdlets.
The Get-BrokerServiceConfigurationData cmdlet shows a list of currently configured settings. A setting that has not been configured has its default value and does not appear in this list. An example setting can be specified as follows:

- Set-BrokerServiceConfigurationData HostingManagement.MaxRegistrationDelayMin -SettingValue 21
  To revert a setting to its default value, using the same example setting as above, do:
- Set-BrokerServiceConfigurationData HostingManagement.MaxRegistrationDelayMin

Core Settings

Core.Heartbeatperiodms

Type: int
Default Value: 600000
Summary:

Controls both the interval and timeouts used for the keep-alive 'pings' from the VDA.

This value is sent from the DDC to VDA and causes the VDA to ping the DDC at an interval half that of the time specified by this setting. By default the DDC will consider contact to have been lost, and discard the VDA's registration, if no ping is received within the full time specified (i.e. the timeout is double the ping interval).

This setting is dynamic, that is, changing it immediately alters both the active ping interval for all VDAs and the maximum interval enforced by the DDC.

The maximum period over which no ping is received before contact is considered to have been lost can be controlled independently of the VDA ping interval itself using the MaxHeartbeatIntervalMs setting.

Core.Maxdisconnectwaittimesecs

Type: int
Default Value: 10
Summary:

Maximum time that the VDA will wait for VDI sessions to disconnect when requested as part of user-driven restart request from StoreFront. This timeout value is sent to the VDA as part of the disconnect request and thus has no impact on the overall request timeout if there are network connectivity issues (see DisconnectOperationTimeOutSecs).

Core.Maxregistrationcompletiontimesecs

Type: int
Default Value: 600
Summary:

Maximum time within which the registration sequence for a single machine must complete. This refers to both immediate hard registrations, and soft to hard registration transitions. If the registration fails to complete within this time then the machine's partial registration is discarded by the broker.

Core.Maxsessionestablishmenttimesecs

Type: int
Default Value: 200
Summary:

Used for logon ticket lifetime, VDA listening timeout, and deadline imposed by the broker for evidence of client connection.

Core.Maxworkers

Type: int
Default Value: 11000
Summary:

The limit for the number of registered VDAs that the controller will accept.

Hostingmanagement Settings

Hostingmanagement.Completedactionretentionperiodsec

Type: int
Default Value: 259200
Summary:

How long a completed power action is retained in the database before being purged.

Hostingmanagement.Maxfailedregistrationsallowed

Type: int
Default Value: 2
Summary:

How many times a VM can fail to register before we put it into maintenance mode. A negative value means that we never automatically put a VM into maintenance mode.

Hostingmanagement.Maxregistrationdelaymin

Type: int
Default Value: 20
Summary:

How long to wait in minutes after a VM is powered on before a failure to receive a registration from the VM is deemed a problem.

This setting is also used in combination with the RebootSchedule/MaxShutdownDelayMin setting to define the maximum allowed time for a machine (either physical or a VM) to successfully reboot during reboot schedule processing.

Logging Settings

Logging.Connectionloglifetimehours

Type: int
Default Value: 48
Summary:

Time for which connection log entries are kept before being purged.

Logging.Hypervisoralertlifetimehours

Type: int
Default Value: 168
Summary:

Time for which hypervisor alert entries are kept before being purged.

Namecache Settings

Namecache.Namerefreshperiodaftererrormins

Type: int
Default Value: 60
Summary:

Starting period after which cached AD user/group account name, or machine name details are refreshed in the case where the SAM name of the cached entity could not be obtained (the cache may thus either contain no SAM name information, or potentially an out of date value). This period is increased exponentially depending on the number of consecutive lookup failures.

Namecache.Namerefreshperiodmins

Type: int
Default Value: 1440
Summary:

Period after which cached AD user/group account name, or machine name details are refreshed in the case where the SAM name of the cached entity was successfully obtained.

See Also

- about_Broker_Concepts
- Get-BrokerServiceConfigurationData
- Set-BrokerServiceConfigurationData

Add-BrokerApplication

March 11, 2024

Adds applications to a desktop group or application group.

Syntax

```
Add-BrokerApplication
   [-InputObject] <Application[]>
   [-ApplicationGroup <ApplicationGroup>]
   [-DesktopGroup <DesktopGroup>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-BrokerApplication
   [-Name] <String>
   [-ApplicationGroup <ApplicationGroup>]
   [-DesktopGroup <DesktopGroup>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-BrokerApplication cmdlet is used to associate one or more applications with an existing desktop group or application group.

There are two parameter sets for this cmdlet, allowing you to specify the application either by its BrowserName or by an array of object references. Uids can also be substituted for the object references.

See about_Broker_Desktops and about_Broker_Applications for more information.

Examples

EXAMPLE 1

Adds the application with a Name of “Notepad”to the desktop group called “Private DesktopGroup”
.


```
Add-BrokerApplication -Name "Notepad" -DesktopGroup "Private
    DesktopGroup"
```

Parameters

-InputObject

Specifies the application to associate. Its Uid can also be substituted for the object reference.

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application to be associated with the desktop group.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ApplicationGroup

Specifies which application group this application should be associated with.

Type:	ApplicationGroup
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Specifies which desktop group this application should be associated with. Note that applications can only be associated with desktop groups of the AppsOnly or DesktopsAndApps delivery type.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Priority

Specifies the priority of the mapping between the application and desktop group where lower numbers imply higher priority with zero being highest.

If one association has a higher priority than the other, machines from that group will be selected for launching sessions until all machines are at maximum load, in maintenance mode, unregistered, or unavailable for any other reason. Only when all machines from the higher-priority group are unavailable will new connections be routed to the next lowest priority group.

If multiple associations with equal priorities are encountered, session launches will be load balanced across all machines in both groups. The least-loaded machine across the groups will be chosen.

This parameter is used only when adding an application to a desktop group. It is an error to specify a priority when adding an application to an application group.

Type:	Int32
Position:	Named

Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Application, or as appropriate by property name

You can pipe the application to be added to Add-BrokerApplication. You can also pipe some of the other parameters by name.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Add-BrokerApplication
- Add-BrokerTag
- Remove-BrokerApplication
- Rename-BrokerApplication
- Move-BrokerApplication
- Set-BrokerApplication

Add-BrokerApplicationGroup

March 11, 2024

Adds application groups to a desktop group.

Syntax

```
Add-BrokerApplicationGroup
   [-InputObject] <ApplicationGroup[]>
   [-DesktopGroup <DesktopGroup>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Add-BrokerApplicationGroup
   [-Name] <String>
   [-DesktopGroup <DesktopGroup>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-BrokerApplicationGroup cmdlet is used to associate one or more application groups with an existing desktop group.

Associating an application group with a desktop group makes the desktop groups's machines available for launching applications in that application group. Until an application group is associated with a desktop group, its applications cannot be launched.

There are two parameter sets for this cmdlet, allowing you to specify the application groups either by Name or by an array of object references. Uids can also be substituted for the object references.

See about_Broker_Applications for more information.

Examples

EXAMPLE 1

Adds the application group with the Name "Office Apps"to the desktop group called "Private DesktopGroup".

```
Add-BrokerApplicationGroup -Name "Office Apps" -DesktopGroup "Private
   DesktopGroup"
```

Parameters

-InputObject

Specifies the application group to associate with the desktop group. Its Uid can also be substituted for the object reference.

Type:	ApplicationGroup[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application group to associate with the desktop group.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroup

Specifies the desktop groups with which the application groups should be associated.

Note that application groups can only be associated with desktop groups whose delivery type is either AppsOnly or DesktopsAndApps.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Priority

Specifies the priority of the mapping between the application group and desktop group. Lower numbers imply higher priority with zero being highest.

If one association has a higher priority than the other, machines from that group will be selected for launching sessions until all machines are at maximum load, in maintenance mode, unregistered, or unavailable for any other reason. Only when all machines from the higher-priority desktop group are unavailable will new connections be routed to the next lowest priority desktop group.

If multiple associations with equal priorities are encountered, session launches will be load balanced across all machines in both desktop groups. The least-loaded machine across the desktop groups will be chosen.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ApplicationGroup, or as appropriate by property name

You can pipe the application group to be added to Add-BrokerApplicationGroup. You can also pipe some of the other parameters by name.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Applications
- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Rename-BrokerApplicationGroup
- Move-BrokerApplicationGroup
- Set-BrokerApplicationGroup

Add-BrokerDesktopGroup

March 11, 2024

Associate Remote PC desktop groups with the specified Remote PC catalog.

Syntax

```
Add-BrokerDesktopGroup
   [-InputObject] <DesktopGroup[]>
   [-RemotePCCatalog <Catalog>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-BrokerDesktopGroup
   [-Name] <String>
   [-RemotePCCatalog <Catalog>]
   [-Priority <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet forms relationships between Remote PC desktop groups and catalogs.

The Remote PC relationships are used by Remote PC automation to determine which desktop groups a machine in a particular Remote PC catalog can be published to. The assignment policy rules belonging to those desktop groups also determines the set of users that are allowed to be assigned to machines from the catalog.

Examples

EXAMPLE 1

Add all Remote PC desktop groups to Remote PC catalog 42.

```
Get-BrokerDesktopGroup -IsRemotePC $true | Add-BrokerDesktopGroup -
   RemotePCCatalog 42
```

EXAMPLE 2

Add desktop groups with names containing MyGroup to Remote PC catalog with name “RPCCat”.

```
Add-BrokerDesktopGroup -Name *MyGroup* -RemotePCCatalog RPCCat
```


Parameters

-InputObject

Specifies one or more Remote PC desktop groups to add to a Remote PC catalog.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the Remote PC desktop groups to add to a Remote PC catalog based on their name properties.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RemotePCCatalog

The Remote PC catalog which the desktop groups are to be added to. Specified by name, Uid or instance.

Type:	Catalog
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Priority

Desktop group to catalog associations carry a priority number, where numerically lower values indicate a higher priority.

The priority relative to other associations determines which desktop group Remote PC automation will move a qualifying unconfigured machine into when it registers. Priority also determines which desktop group a machine will be published to when a user is assigned to the machine by Remote PC automation.

If a value is not supplied, then the desktop group association is automatically assigned a lower priority than any existing associations.

If a priority value is specified that conflicts with an existing association's priority value, then the new association is inserted with that value and existing associations are renumbered upwards to accommodate it.

Type:	Int32
Position:	Named
Default value:	See description
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

The set of Remote PC desktop groups to be added to the catalog can be piped into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_RemotePC
- Move-BrokerDesktopGroup
- Remove-BrokerDesktopGroup
- New-BrokerCatalog
- Remove-BrokerCatalog

Add-BrokerMachine

March 11, 2024

Adds one or more machines to a desktop group.

Syntax

```
Add-BrokerMachine
   [-InputObject] <Machine[]>
   [-DesktopGroup <DesktopGroup>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-BrokerMachine
   [-MachineName] <String>
   [-DesktopGroup <DesktopGroup>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-BrokerMachine cmdlet adds specified machines to a desktop group. There are three forms:

- Use the -InputObject parameter to add a single machine instance or array of instances to the group.
- Use the -MachineName parameter to add a single, named machine to the group.
- Use pipelining to pipe machines instances to the command.

The desktop group to which the machines are added can be specified by name, unique identifier (UID), or instance.

For a machine to be used in a site, the machine must be added to a desktop group. The machine and desktop group must be compatible in order for the process to succeed; for example a machine in a single-session catalog cannot be added to a multi-session desktop group.

For more information about machines, see about_Broker_Machines.

Examples

EXAMPLE 1

These examples all add a single machine instance to a desktop group, identifying the group by instance, UID, or name.

```
Add-BrokerMachine -InputObject $machine -DesktopGroup $desktopGroup
Add-BrokerMachine -InputObject $machine -DesktopGroup 2
Add-BrokerMachine $machine -DesktopGroup "MyDesktopGroup"
```

EXAMPLE 2

These examples add the machine called MyMachine to a desktop group.

```
Add-BrokerMachine -MachineName "MyDomain\MyMachine" -DesktopGroup 2
Add-BrokerMachine "MyDomain\MyMachine" -DesktopGroup "MyDesktopGroup"
Add-BrokerMachine "MyDomain\MyMachine" -DesktopGroup $desktopGroup
```

EXAMPLE 3

These examples find specific machines and add them to a desktop group.

```
Get-BrokerMachine -Uid 3 | Add-BrokerMachine -DesktopGroup 2
Get-BrokerMachine -CatalogUid 4 | Add-BrokerMachine -DesktopGroup 2
```

Parameters

-InputObject

An array of machines to add to the group.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

The name of the single machine to add (must match the MachineName property of the machine).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroup

The desktop group to which the machines are added, specified by name, Uid, or instance.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

You can pipe in the machines you want to add.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Machines
- Add-BrokerMachinesToDesktopGroup
- Remove-BrokerMachine
- Get-BrokerMachine

Add-BrokerMachineConfiguration

March 11, 2024

Adds a machine configuration to a desktop group.

Syntax

```
Add-BrokerMachineConfiguration
   [-InputObject] <MachineConfiguration[]>
   [-Application <Application>]
   [-DesktopGroup <DesktopGroup>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-BrokerMachineConfiguration
   [-Name] <String>
   [-Application <Application>]
   [-DesktopGroup <DesktopGroup>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Associates a machine configuration with a desktop group. The settings in the machine configuration are then applied to the machines in the desktop group.

Examples

EXAMPLE 1

Adds the machine configuration named UPM\Conf1 to the desktop group with Uid 1.

```
Add-BrokerMachineConfiguration -Name UPM\Conf1 -DesktopGroup 1
```

EXAMPLE 2

Adds the machine configuration $mc to the desktop group named "AdminDesktops".

```
$mc | Add-BrokerMachineConfiguration -DesktopGroup AdminDesktops
```


Parameters

-InputObject

Machine configuration to add to the desktop group.

Type:	MachineConfiguration[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Name of a machine configuration to add to the desktop group.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Application

The application to which the machine configurations are added, specified by name, Uid, or instance.

Type:	Application
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DesktopGroup

The desktop group to which the machine configurations are added, specified by name, Uid, or instance.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.MachineConfiguration

The machine configuration to add to the desktop group.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots
- about_Broker_Filtering

Add-BrokerMachinesToDesktopGroup

March 11, 2024

Adds machines from a catalog to a desktop group.

Syntax


```
Add-BrokerMachinesToDesktopGroup
   [-Catalog] <Catalog>
   [-DesktopGroup] <DesktopGroup>
   [-Count] <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-BrokerMachinesToDesktopGroup cmdlet adds a specified number of machines from a catalog to a desktop group.

The cmdlet adds as many machines as possible from the catalog to the desktop group, up to the specified number. The number of machines successfully added to the desktop group is returned.

The machines are added randomly from the catalog and are selected from those that are not already members of a desktop group, and not already assigned to a client, IP address, or user.

Both the catalog and desktop group can be referenced either by instance, name, or unique identifier (Uid). The allocation type of the catalog must be compatible with the type of desktop group. This means the session support (single/multi) and the allocation type (private/shared) of the catalog must match the session support and allocation type in the desktop group.

Examples

EXAMPLE 1

All these examples request that a thousand machines from a catalog are added to a desktop group. The first example references both catalog and desktop group by instance. The second example references both catalog and desktop group by name. The third example references both catalog and desktop group by Uid.

```
Add-BrokerMachinesToDesktopGroup -Catalog $catalog -DesktopGroup
   $desktopGroup -Count 1000
Add-BrokerMachinesToDesktopGroup -Catalog "MyCatalog" -DesktopGroup "
   MyDesktopGroup" -Count 1000
Add-BrokerMachinesToDesktopGroup -Catalog 23 -DesktopGroup 4 -Count
   1000
```

EXAMPLE 2

This example takes ten machines from each manually provisioned catalog and adds them to the specified desktop group.


```
Get-BrokerCatalog -ProvisioningType Manual | Add-
    BrokerMachinesToDesktopGroup -DesktopGroup $desktopGroup -Count 10
```

Parameters

-Catalog

The catalog from which the machines are taken.

Type:	Catalog
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

The desktop group to which the machines are added.

Type:	DesktopGroup
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Count

The number of machines to add to the desktop group.

Type:	Int32
Position:	4

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

You can pipe in the catalog from which the machines are taken.

Int32

You can pipe in the Uid of the catalog from which the machines are taken.

String

You can pipe in the name of the catalog from which the machines are taken.

Outputs

Int32

The number of machines added to the desktop group.

Related Links

- about_Broker_Desktops
- about_Broker_Machines
- Add-BrokerMachine
- Remove-BrokerMachine

Add-BrokerScope

March 11, 2024

Add the specified catalog/desktop group to the given scope(s).

Syntax

```
Add-BrokerScope
   [-InputObject] <Scope[]>
   [-ApplicationGroup <ApplicationGroup>]
   [-Catalog <Catalog>]
   [-DesktopGroup <DesktopGroup>]
   [-PolicySet <GpoPolicySet>]
   [-Tag <Tag>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

The Add-BrokerScope cmdlet is used to associate a scopeable object with the given scope(s). A scopeable object is one of:

- a catalog
- a desktop group
- an application group

To add a scopeable object to a scope you need permission to change the scopes of the scopeable object and permission to add objects to all of the scopes you have specified.

If the scopeable object is already in any scope supplied, that scope will be silently ignored.

Examples

EXAMPLE 1

Adds the desktop group with Uid 27 to the Sales scope.

```
1 Add-BrokerScope Sales -DesktopGroup 27
```

EXAMPLE 2

Adds the BangaloreMachines catalog to the scope with the specified ScopeID.

```
1 Add-BrokerScope BFC74867-C6EF-482C-996F-3E0D340E96AC -Catalog
     BangaloreMachines
```

Parameters

-InputObject

Specifies the scope(s) to add the object to. Each can take the form of either the string form of the scope's GUID or its name.

Type:	Scope[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ApplicationGroup

Specifies the application group to be added. This can take the form of an existing application group object, an application group Uid or name.

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Catalog

Specifies the catalog object to be added. This can take the form of an existing catalog object, a catalog Uid or name.

Type:	Catalog
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Specifies the desktop group object to be added. This can take the form of an existing desktop group object, a desktop group Uid or name.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PolicySet

Specifies the policy set object to be added. This can take the form of an existing policy set object, a policy set Uid or name.

Type:	GpoPolicySet
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Tag

Specifies the tag object to be added. This can take the form of an existing tag object, a tag Uid or name.

Type:	Tag
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Scope

You can pipe scopes to Add-BrokerScope.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Add-BrokerStorefrontAddress

March 11, 2024

Modifies a StoreFront address configuration by adding an additional StoreFront address to it.

Syntax

```
Add-BrokerStorefrontAddress
   [-ByteArray] <Byte[]>
   -Name <String>
   -Url <String>
   -Enabled <Boolean>
   -Description <String>
   [<CommonParameters>]
```

Description

Use this command to transform an existing StoreFront configuration into a new configuration, where the new configuration contains one additional address. The original configuration is supplied as an input, along with the properties of the new StoreFront address being added. The cmdlet outputs the modified configuration, which can then be passed to the Citrix Broker Service using the Add-BrokerMachineConfiguration command.

This command does not, by itself, have any persistent effects within XenDesktop. To make the change persistent, the new configuration byte array must first be transformed into a machine configuration within the Citrix Broker Service. To do this, use the New-BrokerMachineConfiguration command. You can then use the Add-BrokerMachineConfiguration and Set-BrokerMachineConfiguration commands to fully associate the new configuration with a delivery group.

Examples

EXAMPLE 1

This command transforms the configuration byte array specified by $originalConfiguration, adds the new StoreFront details, and stores the resulting configuration in $newConfiguration.


```
$newConfiguration = Add-BrokerStorefrontAddress -ByteArray
    $originalConfiguration -Url "https://mysite.com/Citrix/StoreWeb" -
    Description "This StoreFront delivers my corporate applications" -
    Name "StoreFront1" -Enabled $true
```

Parameters

-ByteArray

Specifies the initial configuration, on which the new configuration is based. All of the addresses in the original configuration are also present in the new configuration, along with the additional address specified. This configuration byte array is obtained from an earlier call to New-BrokerStorefrontAddress, or from Get-BrokerMachineConfiguration.

Type:	Byte[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the new StoreFront.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Url

Specifies the URL to the StoreFront, such as “https://mysite.com/Citrix/StoreWeb”.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Specifies if the new StoreFront address should be enabled for user access.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Specifies a human-readable description of the new StoreFront.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Byte[]

This cmdlet accepts configurations as pipeline input, as an alternative to supplying the ByteArray parameter.

Outputs

Byte[]

This cmdlet outputs the new, modified configuration. This differs from the original configuration in that it contains the additional StoreFront address.

Related Links

- New-BrokerStorefrontAddress
- Get-BrokerStorefrontAddress
- New-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration

Add-BrokerTag

March 11, 2024

Associate a tag with another object in the site.

Syntax


```
Add-BrokerTag
   [-InputObject] <Tag[]>
   [-Application <Application[]>]
   [-ApplicationGroup <ApplicationGroup[]>]
   [-Catalog <Catalog[]>]
   [-Desktop <Desktop[]>]
   [-DesktopGroup <DesktopGroup[]>]
   [-Machine <Machine[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-BrokerTag
   [-Name] <String>
   [-Application <Application[]>]
   [-ApplicationGroup <ApplicationGroup[]>]
   [-Catalog <Catalog[]>]
   [-Desktop <Desktop[]>]
   [-DesktopGroup <DesktopGroup[]>]
   [-Machine <Machine[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Associates one or more tags with a machine, application group, application, desktop group or desktop within the site.

Examples

EXAMPLE 1

Associates ‘Tag1’with machine DOMAIN\Machine.

```
Add-BrokerTag -Name 'Tag1' -Machine DOMAIN\Machine
```

EXAMPLE 2

Creates a new tag with name ‘Tag2’and associates it with the machine with Uid 1.

```
$machine = Get-BrokerMachine -Uid 1
New-BrokerTag 'Tag2' | Add-BrokerTag -Machine $machine
```


Parameters

-InputObject

Specifies one or more tag objects.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies a tag by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Application

Associates the tag with the specified application.

Type:	Application[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-ApplicationGroup

Associates the tag with the specified application group.

Type:	ApplicationGroup[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Catalog

Associates the tag with the specified catalog.

Type:	Catalog[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Desktop

Associates the tag with the specified desktop. The tag is associated with the underlying machine not with the desktop itself.

This parameter is deprecated, use -Machine instead.

Type:	Desktop[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Associates the tag with the specified desktop group.

Type:	DesktopGroup[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Machine

Associates the tag with the specified machine.

Type:	Machine[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Tag

Tags may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerTag

- New-BrokerTag
- Remove-BrokerTag
- Rename-BrokerTag
- Set-BrokerTag

Add-BrokerUser

March 11, 2024

Creates an association between a user and another broker object

Syntax

```
Add-BrokerUser
    [-InputObject] <User[]>
    [-ApplicationGroup <ApplicationGroup>]
    [-Application <Application>]
    [-SessionLinger <SessionLinger>]
    [-SessionPreLaunch <SessionPreLaunch>]
    [-Machine <Machine>]
    [-PrivateDesktop <PrivateDesktop>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-BrokerUser
    [-Name] <String>
    [-ApplicationGroup <ApplicationGroup>]
    [-Application <Application>]
    [-SessionLinger <SessionLinger>]
    [-SessionPreLaunch <SessionPreLaunch>]
    [-Machine <Machine>]
    [-PrivateDesktop <PrivateDesktop>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Add-BrokerUser cmdlet adds broker user objects to another specified object, such as a broker private desktop. This depends on the target object type:

- Machine - assign the broker machine to the specified user(s); when the

machine is subsequently added to a desktop group, the desktop is also assigned to the same user(s).

- PrivateDesktop - assign the desktop to the specified user(s).
- Application - assign the application to the specified user(s).
- Application group - assign the applications in the application group to the specified user(s).

Examples

EXAMPLE 1

Assign the specified private desktop to the specified user.

```
1  Add-BrokerUser "DOMAIN\UserName" -PrivateDesktop "DOMAIN\MachineName"
```

EXAMPLE 2

Assign the specified application to the specified user.

```
1  Add-BrokerUser "DOMAIN\UserName" -Application "ApplicationName"
```

Parameters

-InputObject

The user objects to add.

Type:	User[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the user or users to be added.

Type:	String
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ApplicationGroup

The application group to which the user is to be assigned.

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Application

The application to which the user is to be assigned.

Type:	Application
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionLinger

The session linger setting to which the user is to be assigned.

Type:	SessionLinger
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionPreLaunch

The session pre-launch setting to which the user is to be assigned.

Type:	SessionPreLaunch
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Machine

The machine to which the user is to be assigned

Type:	Machine
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PrivateDesktop

The desktop to which the user is to be assigned

Type:	PrivateDesktop
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.USer

You can pipe the users to be added to Add-BrokerUser.

Outputs

None

By default, this cmdlet returns no output.

Notes

Specify one of the -Machine or -PrivateDesktop or -Application parameters only.

Related Links

- Get-BrokerUser
- Remove-BrokerUser

Copy-BrokerGpoPolicy

March 11, 2024

Copy policies to other policy sets.

Syntax

```
Copy-BrokerGpoPolicy
    -PolicyGuids <Guid[]>
    -ToPolicySets <Guid[]>
    [-WithFilters]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Copy policies, which must be from the same policy set, to other policy sets. The names of the policies to be copied must not exist in the destination policy sets. Priorities of the policies in the destination sets may not be in the same order as the original policies.

Examples

Parameters

-PolicyGuids

The GUIDs of the policies to be copied.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ToPolicySets

The GUIDs of the destination policy sets. The GUID of the policy set that contains the source policies must not be in this array.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-WithFilters

Indicates that filters of each policy are to be copied.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Copy-BrokerGpoPolicySet

March 11, 2024

Make a full or partial copy of a policy set

Syntax

```
Copy-BrokerGpoPolicySet
    -PolicySetGuid <Guid>
    -Name <String>
    [-WithFilters]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Copy a policy set, including the policies, and the settings and filters in each policy. The type of the new policy set is always DeliveryGroupPolicies. If errors are encountered during copying, the resulting policy set may contain incomplete data.

Examples

EXAMPLE 1

Copy a policy set.

```
Copy-BrokerGpoPolicySet -Name newSet -PolicySetGuid "1234..."
```

Parameters

-PolicySetGuid

The GUID of the policy set to be copied

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the new policy set

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-WithFilters

Indicate if filters should be copied

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicySet

Copy-BrokerGpoPolicySet returns the new policy set object.

Related Links

- about_Broker_GroupPolicy
- New-BrokerGpoPolicySet
- Set-BrokerGpoPolicySet
- Remove-BrokerGpoPolicySet

Disable-BrokerGpoPolicy

March 11, 2024

Disable a collection of policies, which must be in the same set.

Syntax

```
Disable-BrokerGpoPolicy
        [-PolicyGuids] <Guid[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Examples

EXAMPLE 1

Disable the policies.


```
Disable-BrokerGpoPolicy -PolicyGuids @("abcdef12-...", "12345678-...")
```

Parameters

-PolicyGuids

GUIDs of the policies to be disabled.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Enable-BrokerGpoPolicy

Disconnect-BrokerSession

March 11, 2024

Disconnect a session.

Syntax

```
Disconnect-BrokerSession
           [-InputObject] <Session[]>
           [-LoggingId <Guid>]
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```


Description

Disconnects the specified session.

If the session is active, no warning is issued to the user before that session is disconnected.

After disconnection, sessions enter a Disconnected state. In a Disconnected state, a session still exists but there is no remote connection to that session.

Examples

EXAMPLE 1

Attempts to disconnect all of the sessions for the user MyDomain\MyAccount.

```
Get-BrokerSession -UserName MyDomain\MyAccount | Disconnect-
    BrokerSession
```

EXAMPLE 2

Disconnects the session on MyMachine.

```
$desktop = Get-BrokerDesktop -DNSName MyMachine.MyDomain.com
Disconnect-BrokerSession $desktop.SessionUid
```

EXAMPLE 3

Attempts to disconnect sessions 10, 11 and 12. Traps and displays the error information.

```
trap  [Citrix.Broker.Admin.SDK.SdkOperationException]
{

  write $("Exception name = " + $_.Exception.GetType().FullName)
  write $("SdkOperationException.Status = " + $_.Exception.Status)
  write $("SdkOperationException.ErrorData=")
  $_.Exception.ErrorData

  write $("SdkOperationException.InnerException = " + $_.Exception.
      InnerException)
  $_.Exception.InnerException
  continue
 }


Disconnect-BrokerSession -InputObject 10,11,12
```


Parameters

-InputObject

Identifies the session(s) to disconnect. This can be expressed as either a session Uid or a session object.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The sessions to disconnect can be piped into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the controller and the machine, or if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Disconnect-BrokerSession could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the disconnect session operation is invoked on only the valid sessions.

The system can fail to disconnect the session if the machine is not in an appropriate state or if there are problems in communicating with the machine. When a disconnect is requested the system detects if the operation was initiated successfully or not by the machine. As this operation is non-blocking the system doesn't detect or report whether the disconnect ultimately succeeded or failed after it was started.

Disconnect failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandling). In the event of errors the SdkErrorRecord error status is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.

- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked on the machine.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed by the machine.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:

"Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:<OperationsSucceededCount>"

Related Links

- Get-BrokerSession
- Stop-BrokerSession

Enable-BrokerGpoPolicy

March 11, 2024

Enable a collection of policies, which must be in the same set.

Syntax

```
Enable-BrokerGpoPolicy
       [-PolicyGuids] <Guid[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Examples

EXAMPLE 1

Enable the policies.

```
Enable-BrokerGpoPolicy -PolicyGuids @("abcdef12-...", "12345678-...")
```


Parameters

-PolicyGuids

GUIDs of the policies to be enabled.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Disable-BrokerGpoPolicy

Export-BrokerConfiguration

March 11, 2024

Obtains an XML document containing the configuration of the broker and optionally a script to import it into another broker

Syntax

```
Export-BrokerConfiguration
      [-TargetBrokerVersion <Version>]
      [-ExistingImportScriptId <String>]
      [-ExistingConfigLastChangeTime <String>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Examples

Parameters

-TargetBrokerVersion

The version of the broker receiving the configuration

Type:	Version
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExistingImportScriptId

The Id of the script the caller already has; a new script will be returned in the XML document if different.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExistingConfigLastChangeTime

The value of ConfigLastChangeTime in the site object of any configuration already held by the caller. If nothing has changed, an empty configuration will be returned with the “Updated”attribute set to false.

Type:	String
Position:	Named

Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet does not accept input

Outputs

String

The XML document

Related Links

Export-BrokerDesktopPolicy

March 11, 2024

Gets the policies of a policy set as a byte array.

Syntax

```
Export-BrokerDesktopPolicy
      [-PolicySetGuid <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Export-BrokerDesktopPolicy returns an array of bytes containing the policies in a policy set.

Examples

EXAMPLE 1

This command exports the policy settings of the policy set.

```
$policy = Export-BrokerDesktopPolicy
```

Parameters

-PolicySetGuid

The GUID of the policy set to export. If this is not specified, the data of the site default policy set is exported.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

The configuration data as an opaque binary blob. This will be null if the policy set contains no data.

Related Links

- Import-BrokerDesktopPolicy
- New-BrokerConfigurationSlot
- New-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

Export-BrokerPiiData

March 11, 2024

Exports the personal identifation information and scrambled values

Syntax

```
Export-BrokerPiiData
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Export-BrokerPiiData exports the personal identification information (PII) data and scrambled values. If the admin chooses to send scrambled PII data in the licensing events, the original data and its scrambled values can be exported by this cmdlet.

The retention time of the PII data is one year from the last accessed date of the specific PII data.

Examples

EXAMPLE 1

Returns all the PII data and the scrambled values and save the output

to the target file.

```
1  Export-BrokerPiiData | Out-File -Path C:\Foo\data.csv
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Byte[]

Export-BrokerPiiData returns a list of PII data and scrambled values

Related Links

Export-BrokerPolicyTemplates

March 11, 2024

Gets the site wide Citrix Group Policy templates.

Syntax

```
Export-BrokerPolicyTemplates
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Export-BrokerPolicyTemplates returns an array of bytes containing the site-wide Citrix Group Policy templates. These templates can be used to create new policies.

Examples

EXAMPLE 1

This command exports the site wide Citrix Group Policy templates.

```
$policy = Export-BrokerPolicyTemplates
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

The policy templates data as an opaque binary blob. This will be null if no site wide Citrix Group Policy templates have been created.

Notes

Export-BrokerPolicyTemplates performs a specialized operation. Direct usage of it in scripts is discouraged, and could result in data corruption. It is recommended that this operation only be performed via the Citrix Studio.

Related Links

- Import-BrokerPolicyTemplates

Get-BrokerAccessPolicyRule

March 11, 2024

Gets rules from the site's access policy.

Syntax

```
Get-BrokerAccessPolicyRule
   [[-Name] <String>]
   [-AllowedConnections <AllowedConnection>]
   [-AllowedUsers <AllowedUser>]
   [-Description <String>]
   [-DesktopGroupName <String>]
   [-DesktopGroupUid <Int32>]
   [-Enabled <Boolean>]
```



```
    [-ExcludedClientIPFilterEnabled <Boolean>]
    [-ExcludedClientName <String>]
    [-ExcludedClientNameFilterEnabled <Boolean>]
    [-ExcludedSmartAccessFilterEnabled <Boolean>]
    [-ExcludedSmartAccessTag <String>]
    [-ExcludedUser <User>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-IncludedClientIPFilterEnabled <Boolean>]
    [-IncludedClientName <String>]
    [-IncludedClientNameFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterType <String>]
    [-IncludedSmartAccessTag <String>]
    [-IncludedUser <User>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerAccessPolicyRule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns rules matching the specified search criteria from the site's access policy. If no search criteria are specified, all rules in the access policy are obtained.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

———————BrokerAccessPolicyRule Object

A BrokerAccessPolicyRule object represents a single rule within the site's access policy. For a user to gain access to a desktop group via the rule their connection must match all its enabled include filters, and none of its enabled exclude filters. The object contains the following properties:

- AllowedConnections (Citrix.Broker.Admin.SDK.AllowedConnection)

Controls whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.
For a detailed description of this property see "help about_Broker_AccessPolicy".

- AllowedProtocols (System.String[])

  Protocols (for example HDX, RDP) available to the user for sessions delivered from the rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.
  If the protocol list is empty, access to the desktop group is implicitly denied.

- AllowedUsers (Citrix.Broker.Admin.SDK.AllowedUser)

  Controls the behavior of the included users filter. This can restrict access to a list of named users or groups, or allow access to any authenticated user. For a detailed description of this property see "help about_Broker_AccessPolicy".

- AllowRestart (System.Boolean)

  Indicates if the user can restart sessions delivered from the rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

- AppProtectionKeyLoggingRequired (System.Boolean)

  Specifies whether key logging app protection is required.

- AppProtectionScreenCaptureRequired (System.Boolean)

  Specifies whether screen capture app protection is required.

- Description (System.String)

  An optional description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

- DesktopGroupName (System.String)

  The name of the desktop group to which the rule applies.

- DesktopGroupUid (System.Int32)

  The unique ID of the desktop group to which the rule applies.

- Enabled (System.Boolean)

  Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's access policy.

- ExcludedClientIPFilterEnabled (System.Boolean)

  Indicates whether the excluded client IP filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- ExcludedClientIPs (Citrix.Broker.Admin.SDK.ChbIPAddressRange[])

  IP addresses of user devices explicitly denied access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the excluded client IP address filter.

- ExcludedClientNameFilterEnabled (System.Boolean)

  Indicates whether the excluded client name filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- ExcludedClientNames (System.String[])

  Names of user devices explicitly denied access to the rule's desktop group. This property forms part of the excluded client names filter.

- ExcludedSmartAccessFilterEnabled (System.Boolean)

  Indicates whether the excluded SmartAccess tags filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- ExcludedSmartAccessTags (System.String[])

  SmartAccess tags which explicitly deny access to the rule's desktop group if any occur in those provided by with the user's connection. This property forms part of the excluded SmartAccess tags filter.

- ExcludedUserFilterEnabled (System.Boolean)

  Indicates whether the excluded users filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- ExcludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  Users and groups who are explicitly denied access to the rule's desktop group. This property forms part of the excluded users filter.

- HdxSslEnabled (System.Boolean)

  Indicates whether TLS encryption is enabled for sessions delivered from the rule's desktop group.

- IncludedClientIPFilterEnabled (System.Boolean)

  Indicates whether the included client IP filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- IncludedClientIPs (Citrix.Broker.Admin.SDK.ChbIPAddressRange[])

  IP addresses of user devices allowed access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

- IncludedClientNameFilterEnabled (System.Boolean)

  Indicates whether the included client names filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- IncludedClientNames (System.String[])

  Names of user devices allowed access to the rule's desktop group. This property forms part of the included client names filter.

- IncludedSmartAccessFilterEnabled (System.Boolean)

  Indicates whether the included SmartAccess tags filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- IncludedSmartAccessFilterType (System.String)

  Indicates whether all tags present in IncludedSmartAccessTags must match tags provided by the user's connection to grant access (MatchAll), or whether any tag matching is sufficient (MatchAny).

- IncludedSmartAccessTags (System.String[])

  The SmartAccess tags which grant access to the rule's desktop group if they occur in those provided with the user's connection. If multiple tags are specified, access also depends on the IncludedSmartAccessFilterType setting. This property forms part of the included SmartAccess tags filter.

- IncludedUserFilterEnabled (System.Boolean)

  Indicates whether the included users filter is enabled. If the filter is disabled it is ignored when the rule is evaluated.

- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  Users and groups who are granted access to the rule's desktop group. This property forms part of the included users filter.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.

- Name (System.String)

  Administrative name of the rule. Each rule in the site's access policy must have a unique name.

- Uid (System.Int32)

  Unique ID of the rule itself.

Examples

EXAMPLE 1

Returns all access policy rules. This offers a complete description of the current site's access policy.

```
Get-BrokerAccessPolicyRule
```

EXAMPLE 2

Returns all rules that are both enabled and explicitly include the SALES\tech-support group in their included users filter.

```
Get-BrokerAccessPolicyRule -Enabled $true -IncludedUser sales\tech-
    support
```

Parameters

-Uid

Gets only the rule with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only rules with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AllowedConnections

Gets only rules that have the specified value in the AllowedConnections property of their included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, ViaAG and AnyViaAG.

Type:	AllowedConnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowedUsers

Gets only rules that have the specified value in the AllowedUsers property of their included users filter.

Valid values are Filtered, AnyAuthenticated, Any, AnonymousOnly and FilteredOrAnonymous.

Type:	AllowedUser
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Description

Gets only rules with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupName

Gets only rules applying to desktop groups with names matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only rules that apply to the desktop group with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only rules that are in the specified state, either enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedClientIPFilterEnabled

Gets only rules that have their excluded client IP address filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedClientName

Gets only rules that have the specified client name in their excluded client names filter (whether the filter is enabled or not).

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExcludedClientNameFilterEnabled

Gets only rules that have their excluded client name filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedSmartAccessFilterEnabled

Gets only rules that have their excluded SmartAccess tags filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedSmartAccessTag

Gets only rules that have the specified SmartAccess tag in their excluded SmartAccess tags filter (whether the filter is enabled or not).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExcludedUser

Gets only rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Gets only rules that have their excluded user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientIPFilterEnabled

Gets only rules that have their included client IP address filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientName

Gets only rules that have the specified user device name in their included client names filter (whether the filter is enabled or not).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IncludedClientNameFilterEnabled

Gets only rules that have their included client name filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessFilterEnabled

Gets only rules that have their included SmartAccess tags filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessFilterType

Gets only rules that have the specified included SmartAccess tags filter type (MatchAll, or MatchAny).

Type:	String
Accepted values:	MatchAll, MatchAny
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessTag

Gets only rules that have the specified SmartAccess tag in their included SmartAccess tags filter (whether the filter is enabled or not).

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-IncludedUser

Gets only rules that have the specified user in their included users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Gets only rules that have their included user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AccessPolicyRule

Get-BrokerAccessPolicyRule returns all access policy rules that match the specified selection criteria.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- New-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule

Get-BrokerAdminFolder

March 11, 2024

Get the admin folders in this site.

Syntax

```
Get-BrokerAdminFolder
    [[-Name] <String>]
    [-DirectChildAdminFolders <Int32>]
    [-DirectChildApplicationGroups <Int32>]
    [-DirectChildApplications <Int32>]
    [-DirectChildCatalogs <Int32>]
    [-DirectChildDesktopGroups <Int32>]
    [-FolderName <String>]
    [-LastChangeId <Guid>]
    [-Metadata <String>]
    [-ParentAdminFolderUid <Int32>]
    [-TotalChildApplicationGroups <Int32>]
    [-TotalChildApplications <Int32>]
    [-TotalChildCatalogs <Int32>]
    [-TotalChildDesktopGroups <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerAdminFolder
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerAdminFolder cmdlet gets admin folders in this site.

Without parameters, Get-BrokerAdminFolder gets all the admin folders that have been created. You can also use the parameters of Get-BrokerAdminFolder to filter the results to just the folders you're interested in. You can also identify folders by their UIDs or their FolderNames.

————————BrokerAdminFolder Object

A folder for use in the administration console for organising other objects. E.g. BrokerApplication objects

- DirectChildAdminFolders (System.Int32)

  The number of admin folders with this folder as a direct parent

- DirectChildApplicationGroups (System.Int32)

  The number of application groups in this admin folder (does not include any application groups in child folders)

- DirectChildApplications (System.Int32)

  The number of applications in this admin folder (does not include any applications in child folders)

- DirectChildCatalogs (System.Int32)

  The number of catalogs in this admin folder (does not include any catalogs in child folders)

- DirectChildDesktopGroups (System.Int32)

  The number of desktop groups in this admin folder (does not include any desktop groups in child folders)

- FolderName (System.String)

  The simple name of this folder within any parent folder

- LastChangeId (System.Guid)

  A random GUID assigned whenever there is a change anywhere in the hierarchy of admin folders below this node; each change updates this value on the changed folder and all parents all the way up to the root folder. Note that nodes below any change do not have their LastChangeId value updated

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Holds any metadata associated with the admin folder

- Name (System.String)

  The full name of the folder including the full parent hierarchy separated by back-slash characters and including a trailing back-slash

- ParentAdminFolderUid (System.Int32)

  The UID of the parent admin folder; the root folder references itself (zero)

- TotalChildApplicationGroups (System.Int32)

  The number of application groups in this admin folder (including any application groups in child folders)

- TotalChildApplications (System.Int32)

  The number of applications in this admin folder (including any applications in child folders)

- TotalChildCatalogs (System.Int32)

  The number of catalogs in this admin folder (including any catalogs in child folders)

- TotalChildDesktopGroups (System.Int32)

  The number of desktop groups in this admin folder (including any desktop groups in child folders)

- Uid (System.Int32)

  The unique ID of the admin folder (the root folder has the value zero)

Examples

EXAMPLE 1

Return all administration folders.

```
Get-BrokerAdminFolder
```

EXAMPLE 2

Get the administration folder with Uid 1.

```
Get-BrokerAdminFolder -Uid 1
```

Parameters

-Uid

Gets only the admin folder with the specified unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets admin folders matching the specified name (if no trailing backslash is supplied, it is assumed).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DirectChildAdminFolders

Gets admin folders with the specified number of child folders.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DirectChildApplicationGroups

Gets admin folders with the specified number of application groups (excluding those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DirectChildApplications

Gets admin folders with the specified number of applications (excluding those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DirectChildCatalogs

Gets admin folders with the specified number of catalogs (excluding those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DirectChildDesktopGroups

Gets admin folders with the specified number of desktop groups (excluding those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FolderName

Gets only the admin folders matching the specified simple folder name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastChangeId

Gets only the admin folders with the specified value for LastChangeId.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ParentAdminFolderUid

Gets only admin folders with the specified parent admin folder UID value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalChildApplicationGroups

Gets admin folders with the specified number of application groups (including those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalChildApplications

Gets admin folders with the specified number of applications (including those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalChildCatalogs

Gets admin folders with the specified number of catalogs (including those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalChildDesktopGroups

Gets admin folders with the specified number of desktop groups (including those in sub-folders).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AdminFolder

Returns admin folders.

Related Links

- about_Broker_Filtering
- New-BrokerAdminFolder
- Remove-BrokerAdminFolder

Get-BrokerAppAssignmentPolicyRule

March 11, 2024

Gets application rules from the site's assignment policy.

Syntax

```
Get-BrokerAppAssignmentPolicyRule
   [[-Name] <String>]
   [-Description <String>]
   [-DesktopGroupUid <Int32>]
   [-Enabled <Boolean>]
   [-ExcludedUser <User>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-IncludedUser <User>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerAppAssignmentPolicyRule
   [-Uid] <Int32>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns application rules matching the specified search criteria from the site's assignment policy. If no search criteria are specified, all application rules in the assignment policy are obtained.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

——————————BrokerAppAssignmentPolicyRule Object

The BrokerAppAssignmentPolicyRule object represents a single application rule within the site's assignment policy. It contains the following properties:

- Description (System.String)

  An optional description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

- DesktopGroupUid (System.Int32)

  The unique ID of the desktop group to which the rule applies.

- Enabled (System.Boolean)

  Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's assignment policy.

- ExcludedUserFilterEnabled (System.Boolean)

  Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

- ExcludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule's desktop group.

- IncludedUserFilterEnabled (System.Boolean)

  Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly entitled to the machine assignment described by the rule.

- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The included users filter of the rule, that is, the users and groups who are entitled to a machine assignment from the rule's desktop group.

- Name (System.String)

  The administrative name of the rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

- Uid (System.Int32)

  The unique ID of the rule itself.

Examples

EXAMPLE 1

Returns all application rules from the assignment policy. This offers a complete description of the current site's assignment policy with respect to machine assignment entitlements for delivery of application sessions from private desktop groups.

```
Get-BrokerAppAssignmentPolicyRule
```

EXAMPLE 2

Returns the rule in the assignment policy that gives users entitlements to machine assignments in the Sales Support desktop group for delivery of application sessions.

```
$dg = Get-BrokerDesktopGroup 'Sales Support'
Get-BrokerAppAssignmentPolicyRule -DesktopGroupUid $dg.Uid
```

Parameters

-Uid

Gets the application rule with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only application rules with the specified name.

Type:	String
Position:	2

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets only application rules with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only application rules that apply to the desktop group with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only application rules that are in the specified state, either enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUser

Gets only application rules that have the specified user in their excluded users filter (whether the filter is enabled or not)

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Gets only application rules that have their excluded user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUser

Gets only application rules that have the specified user in their included users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Gets only application rules that have their included user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

Get-BrokerAppAssignmentPolicyRule returns all application rules in the assignment policy that match the specified selection criteria.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- New-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule
- Remove-BrokerAppAssignmentPolicyRule

Get-BrokerAppEntitlementPolicyRule

March 11, 2024

Gets application rules from the site's entitlement policy.

Syntax

```
Get-BrokerAppEntitlementPolicyRule
    [[-Name] <String>]
    [-Description <String>]
    [-DesktopGroupUid <Int32>]
    [-Enabled <Boolean>]
    [-ExcludedUser <User>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-IncludedUser <User>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-LeasingBehavior <LeasingBehavior>]
    [-SessionReconnection <SessionReconnection>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerAppEntitlementPolicyRule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Returns application rules matching the specified search criteria from the site's entitlement policy. If no search criteria are specified, all application rules in the entitlement policy are obtained.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

——————————BrokerAppEntitlementPolicyRule Object

The BrokerAppEntitlementPolicyRule object represents a single application rule within the site's entitlement policy. It contains the following properties:

- Description (System.String)

  Optional description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

- DesktopGroupUid (System.Int32)

  The unique ID of the desktop group to which the rule applies.

- Enabled (System.Boolean)

  Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

- ExcludedUserFilterEnabled (System.Boolean)

  Indicates whether the excluded users filter of the rule is enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

- ExcludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to use published applications from the associated desktop group.

- IncludedUserFilterEnabled (System.Boolean)

  Indicates whether the included users filter of the rule is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the rule.

- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule.
  If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

- LeasingBehavior (Citrix.Broker.Admin.SDK.LeasingBehavior)

  Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:
  Allowed and Disallowed.

- Name (System.String)

  The administrative name of the rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

- SessionReconnection (Citrix.Broker.Admin.SDK.SessionReconnection)

  Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:
  Always, DisconnectedOnly, and SameEndpointOnly.

- Uid (System.Int32)

  The unique ID of the rule itself.

Examples

EXAMPLE 1

Returns all application rules from the entitlement policy. This offers a complete description of the current site's entitlement policy with respect to application entitlements from shared desktop groups.

```
Get-BrokerAppEntitlementPolicyRule
```

EXAMPLE 2

Returns the application rule in the entitlement policy that grants users an entitlement to application sessions in the Customer Support desktop group.

```
$dg = Get-BrokerDesktopGroup 'Customer Support'
Get-BrokerAppEntitlementPolicyRule -DesktopGroupUid $dg.Uid
```

Parameters

-Uid

Gets the application rule with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only application rules with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets only application rules with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only the application rule that applies to the desktop group with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only application rules that are in the specified state, either enabled ($true), or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUser

Gets only application rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Gets only application rules that have their excluded user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUser

Gets only application rules that have the specified user in their included users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Gets only application rules that have their included user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeasingBehavior

Gets only application rules with the specified connection leasing behavior. Possible values are: Allowed and Disallowed.

Type:	LeasingBehavior
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Gets only application rules with the specified session reconnection behavior. Possible values are: Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

Get-BrokerAppEntitlementPolicyRule returns all application rules from the entitlement policy that match the specified selection criteria.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule

Get-BrokerApplication

March 11, 2024

Get the applications published on this site.

Syntax

```
Get-BrokerApplication
    [[-Name] <String>]
    [-AdminFolderName <String>]
    [-AdminFolderUid <Int32>]
    [-AllAssociatedDesktopGroupUid <Int32>]
    [-AllAssociatedDesktopGroupUUID <Guid>]
    [-ApplicationName <String>]
    [-ApplicationType <ApplicationType>]
    [-AssociatedApplicationGroupUid <Int32>]
    [-AssociatedApplicationGroupUUID <Guid>]
    [-AssociatedDesktopGroupPriority <Int32>]
    [-AssociatedDesktopGroupUid <Int32>]
    [-AssociatedDesktopGroupUUID <Guid>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserSID <String>]
    [-AssociatedUserUPN <String>]
    [-BrowserName <String>]
    [-ClientFolder <String>]
    [-CommandLineArguments <String>]
    [-CommandLineExecutable <String>]
    [-CpuPriorityLevel <CpuPriorityLevel>]
    [-Description <String>]
    [-DoNotEnumerate <Boolean>]
    [-Enabled <Boolean>]
    [-HomeZoneName <String>]
    [-HomeZoneOnly <Boolean>]
    [-HomeZoneUid <Guid>]
```



```
    [-IconFromClient <Boolean>]
    [-IconUid <Int32>]
    [-IgnoreUserHomeZone <Boolean>]
    [-LocalLaunchDisabled <Boolean>]
    [-MaxPerMachineInstances <Int32>]
    [-MaxPerUserInstances <Int32>]
    [-MaxTotalInstances <Int32>]
    [-MetadataKey <String>]
    [-Metadata <String>]
    [-PackagedApplicationId <String>]
    [-PackagedApplicationType <String>]
    [-PublishedName <String>]
    [-SecureCmdLineArgumentsEnabled <Boolean>]
    [-ShortcutAddedToDesktop <Boolean>]
    [-ShortcutAddedToStartMenu <Boolean>]
    [-StartMenuFolder <String>]
    [-Tag <String>]
    [-UserFilterEnabled <Boolean>]
    [-UUID <Guid>]
    [-Visible <Boolean>]
    [-WaitForPrinterCreation <Boolean>]
    [-WorkingDirectory <String>]
    [-DesktopUid <Int32>]
    [-ApplicationGroupUid <Int32>]
    [-SessionUid <Int64>]
    [-UserSID <String>]
    [-DesktopGroupUid <Int32>]
    [-MachineConfigurationUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerApplication
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerApplication cmdlet gets the published applications that are hosted on this site.

Without parameters, Get-BrokerApplication gets all the applications that have been published, regardless of whether they are visible to users or not. You can also use the parameters of Get-BrokerApplication to filter the results to just the applications you're interested in. You can also

identify applications by their UIDs or their BrowserNames.

For more information about applications, see about_Broker_Applications.

----------BrokerApplication Object

The BrokerApplication object represents a published application in the site. It contains the following properties:

- AdminFolderName (System.String)

  The name of the admin folder the application is in (including trailing backslash), or the empty string if the application is at the root level

- AdminFolderUid (System.Int32)

  The Uid of the admin folder the application is in (if any)

- AllAssociatedDesktopGroupUids (System.Int32[])

  List of associated desktop group uids, including desktop groups that are indirectly associated with the application by virtue of being associated with an application group of which this application is a member.

- AllAssociatedDesktopGroupUUIDs (System.Guid[])

  List of associated desktop group UUIDs, including desktop groups that are indirectly associated with the application by virtue of being associated with an application group of which this application is a member.

- ApplicationName (System.String)

  The simple name of the application within its parent admin folder (if any)

- ApplicationType (Citrix.Broker.Admin.SDK.ApplicationType)

  The type of the application, whether HostedOnDesktop, InstalledOnClient or PublishedContent.

- AssociatedApplicationGroupUids (System.Int32[])

  List of associated application group uids.

- AssociatedApplicationGroupUUIDs (System.Guid[])

  List of associated application group UUIDs.

- AssociatedDesktopGroupPriorities (System.Int32[])

  List of directly associated desktop group priorities. Associated desktop groups is the list of desktop groups on which the application is published. When launching an application an available machine from one of the associated groups is selected. Desktop groups are searched for available machines in order of their priority.

- AssociatedDesktopGroupUids (System.Int32[])

  List of directly associated desktop group uids. Associated desktop groups is the list of desktop groups on which the application is published. The list is sorted by priority, with the highest priority group first.

- AssociatedDesktopGroupUUIDs (System.Guid[])

  List of directly associated desktop group UUIDs. Associated desktop groups is the list of desktop groups on which the application is published. The list is sorted by priority, with the highest priority group first.

- AssociatedUserFullNames (System.String[])

  List of associated users (full names). Associated users is the list of users who are given access using the application/user mapping filter.

- AssociatedUserNames (System.String[])

  List of associated users (SAM names). Associated users is the list of users who are given access using the application/user mapping filter.

- AssociatedUserSIDs (System.String[])

  List of associated users (SIDs). Associated users is the list of users who are given access using the application/user mapping filter.

- AssociatedUserUPNs (System.String[])

  List of associated users (user principle names). Associated users is the list of users who are given access using the application/user mapping filter.

- BrowserName (System.String)

  Unique browser name used to identify this application to other components in the site. This value is not visible to the end users.

- ClientFolder (System.String)

  The folder that the application belongs to as the user sees it.

- CommandLineArguments (System.String)

  The command-line arguments to use when launching the executable.

- CommandLineExecutable (System.String)

  The name including the full path of the executable file to launch.

- ConfigurationSlotUids (System.Int32[])

  Uids of any configuration slots which hold machine configurations associated with the application. The order of slot UIDs in this list correspond with the order of items in the associated

MachineConfigurationNames and MachineConfigurationUids list properties, and so the same slot UID can appear more than once.

- CpuPriorityLevel (Citrix.Broker.Admin.SDK.CpuPriorityLevel)

  The CPU priority of the launched process. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High.

- Description (System.String)

  Optional application description. This description is visible to the end users.

- DoNotEnumerate (System.Boolean)

  Specifies if the application is returned to the user by enumeration.

- Enabled (System.Boolean)

  Specifies whether or not this application can be launched.

- HomeZoneName (System.String)

  Name of optional preferred home zone for launching the application.

- HomeZoneOnly (System.Boolean)

  Indicates that if the preferred zone for launching the application is its home zone but no machine is available from that zone then the launch fails.

- HomeZoneUid (System.Guid?)

  Optional preferred home zone for launching the application.

- IconFromClient (System.Boolean)

  Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.

- IconUid (System.Int32?)

  The icon UID used for this application. If not specified a generic icon is used.

- IgnoreUserHomeZone (System.Boolean)

  Indicates that when launching the application and the user has a home zone specified then the user's home zone preference should be ignored.

- LocalLaunchDisabled (System.Boolean)

  When launching a published application from within a published desktop, do not launch the application in that desktop session.

- MachineConfigurationNames (System.String[])

  The MachineConfiguration names associated with the application.

- MachineConfigurationUids (System.Int32[])

  The MachineConfiguration uids associated with the application.

- MaxPerMachineInstances (System.Int32)

  Maximum allowed concurrently running instances of the application that an individual machine can have. A value of zero allows unlimited usage subject to any site-wide limit.

- MaxPerUserInstances (System.Int32)

  Maximum allowed concurrently running instances of the application that an individual user can have. A value of zero allows unlimited usage subject to any site-wide limit.

- MaxTotalInstances (System.Int32)

  Maximum allowed total of concurrently running instances of the application in the site. A value of zero allows unlimited usage.

- MetadataKeys (System.String[])

  All key names of metadata items associated with this application.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this application.

- Name (System.String)

  Unique administrative name of application; this will include any parent admin folder hierarchy separated by backslash characters.

- PackagedApplicationId (System.String)

  The Id of the Packaged Application in the AppLibrary

- PackagedApplicationType (System.String)

  The packaging technology used to create this application

- PublishedName (System.String)

  Published name of application as seen by end user. If not specified value used defaults to the administrative name.

- SecureCmdLineArgumentsEnabled (System.Boolean)

  Specifies whether the command-line arguments should be secured.

- ShortcutAddedToDesktop (System.Boolean)

  Specifies whether a shortcut to the application should be placed on the user device.

- ShortcutAddedToStartMenu (System.Boolean)

  Specifies whether a shortcut to the application should be placed in the user's Start menu on their user device.

- StartMenuFolder (System.String)

  The name of the Start menu folder that holds the application shortcut.

- Tags (System.String[])

  A list of tags associated with the application.

- Uid (System.Int32)

  A unique identifier of the application.

- UserFilterEnabled (System.Boolean)

  Indicates if application-specific user filter is enabled.

- UUID (System.Guid)

  UUID of the application.

- Visible (System.Boolean)

  Specifies if the application is visible to users.

- WaitForPrinterCreation (System.Boolean)

  Specifies whether the VDA delays starting the app until printers are set up or not.

- WorkingDirectory (System.String)

  The working directory the executable is launched from.

Examples

EXAMPLE 1

Returns the application with the Name of “Notepad”.

```
1  Get-BrokerApplication Notepad
```

EXAMPLE 2

Returns the applications that have a PublishedName starting with “Note”and that are enabled.

```
1  Get-BrokerApplication -PublishedName Note* -Enabled $true
```


Parameters

-Uid

Gets only the application with the specified unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only the applications matching the specified name (including any parent admin folder hierarchy.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderName

Gets applications that are in admin folders matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderUid

Gets applications that are in the specified admin folder.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllAssociatedDesktopGroupUid

Gets applications associated with the desktop group identified by the uid.

The application may be either published directly on the desktop group or published indirectly on the desktop group as part of an application group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllAssociatedDesktopGroupUUID

Gets applications associated with the desktop group identified by the UUID.

The application may be either published directly on the desktop group or published indirectly on the desktop group as part of an application group.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationName

Gets applications that match the specified simple name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ApplicationType

Gets applications that match the type specified: HostedOnDesktop, InstalledOnClient or Published-Content.

Type:	ApplicationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedApplicationGroupUid

Gets applications that are members of the application group identified by the uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedApplicationGroupUUID

Gets applications that are members of the application group identified by the UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedDesktopGroupPriority

Gets applications with an associated desktop group identified by priority assigned to the pairing between an application and desktop group.

Associated desktop group is a desktop group on which the application is published.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AssociatedDesktopGroupUid

Gets applications directly associated with the desktop group identified by the uid.

The application must be published directly on the desktop group. To search for applications that may be published indirectly on the desktop group as part of an application group, use the AllAssociated-DesktopGroupUid filter instead.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedDesktopGroupUUID

Gets applications directly associated with the desktop group identified by the UUID.

The application must be published directly on the desktop group. To search for applications that may be published indirectly on the desktop group as part of an application group, use the AllAssociated-DesktopGroupUid filter instead.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserFullName

Gets applications with an associated user identified by their full name (usually 'first-name last-name').

If the 'UserFilterEnabled'property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets applications with an associated user identified by their user name (in the form 'domain\user'). If the 'UserFilterEnabled'property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserSID

Gets applications with an associated user identified by their Windows SID. If the 'UserFilterEnabled' property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserUPN

Gets applications with an associated user identified by their user principle name (in the form 'user@domain'). If the 'UserFilterEnabled'property is true then access to the application is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrowserName

Gets only the applications that match the supplied name. The BrowserName is usually an internal name for the application and is unique in the site.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientFolder

Gets only the applications that match the specified value for the folder the application belongs to as seen by the end-user. This folder can be seen in the Citrix Online Plug-in, in Web Services, and also potentially in the user's start menu.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CommandLineArguments

Gets only the applications that match the supplied arguments to the command-line executable.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CommandLineExecutable

Gets only the applications that match the supplied command-line executable.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-CpuPriorityLevel

Gets only the applications that have the specified value for the CPU priority level of the launched executable. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High.

Type:	CpuPriorityLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets only the applications that match the supplied description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DoNotEnumerate

Gets only the applications that have the specified value for whether the application is hidden from enumeration. Hidden applications are not visible to the user but may still be launched by other means.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only the applications that have the specified value for whether the application is enabled. Disabled applications are still visible to users (that is controlled by the Visible setting) but cannot be launched.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HomeZoneName

Gets only applications with a home zone preference matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HomeZoneOnly

Gets only applications that have the specified behaviour with respect to forcing use of their home zone during launch.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HomeZoneUid

Gets only applications with a home zone preference matching the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconFromClient

Gets only the applications that have the specified value for whether the application icon should be retrieved from the user device.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets only the applications that use the specified icon (identified by its Uid).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreUserHomeZone

Gets only applications that have the specified behaviour with respect to ignoring user home zones during launch.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalLaunchDisabled

Gets only applications that have the specified local launch behaviour

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerMachineInstances

Gets only applications with the specified maximum allowed concurrently running instances that an individual machine can have.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerUserInstances

Gets only applications with the specified maximum allowed concurrently running instances that an individual user can have.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTotalInstances

Gets only applications with the specified maximum allowed total of concurrently running instances in the site.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MetadataKey

Gets only applications whose associated metadata contains key names matching the specified value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PackagedApplicationId

Gets only the applications with the specified Packaged Application Id

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PackagedApplicationType

Gets only the applications with the specified application package type

Type:	String
Accepted values:	AppAttach, AppVDualAdmin, AppVSingleAdmin, FlexApp, Msix, NotApplicable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Gets applications whose published name matches the supplied pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SecureCmdLineArgumentsEnabled

Gets only the applications that have the specified value for whether the command-line arguments should be secured. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShortcutAddedToDesktop

Gets only the applications that match depending on whether a shortcut for the application has been added to the user device or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShortcutAddedToStartMenu

Gets only the applications that match depending on whether a shortcut for the application has been added to Start Menu of the user device or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-StartMenuFolder

Gets only the applications that match the specified name for the start menu folder that holds the application shortcut. This is valid only for the Citrix Online Plug-in.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Tag

Gets applications associated with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserFilterEnabled

Gets only applications whose user filter is in the specified state.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets applications with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Visible

Gets only the applications that have the specified value for whether it is visible to the users.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WaitForPrinterCreation

Gets only the applications that match depending on whether the VDA delays starting the application until printers are set up.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WorkingDirectory

Gets only the applications that match the specified working directory.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopUid

Gets only the applications that have been associated (using a desktop group) to the specified desktop (identified by its Uid). Note that an application is not directly associated with a desktop, but only indirectly by which desktop group it has been published to.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationGroupUid

Gets applications that are members of the application group identified by the uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets only the applications that are running in the specified session (identified by its Uid).

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserSID

Gets only applications with their accessibility restricted to include the specified user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only the applications that have been published to the specified desktop group (identified by its Uid).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineConfigurationUid

Gets only applications which have an associated machine configuration identified by the given Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Application

Get-BrokerApplication returns an object for each application it gets.

Notes

Get-BrokerApplication returns just the application object, and as such is not a complete picture. The returned objects do not tell you what File-Type Associations are configured for this application, etc.

Use the following cmdlets to gather data related to applications (shown with examples of syntax):

Get-BrokerConfiguredFTA -ApplicationUid $app.Uid

Get-BrokerTag -ApplicationUid $app.Uid

Get-BrokerDesktopGroup -ApplicationUid $app.Uid

Get-BrokerDesktop -PublishedApplication $app

Get-BrokerSession -ApplicationUid $app.Uid

Get-BrokerApplicationInstance -ApplicationUid $app.Uid

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Add-BrokerApplication
- Remove-BrokerApplication
- Rename-BrokerApplication
- Move-BrokerApplication
- Set-BrokerApplication

Get-BrokerApplicationGroup

March 11, 2024

Gets details of configured application groups.

Syntax

```
Get-BrokerApplicationGroup
   [[-Name] <String>]
   [-AdminFolderName <String>]
   [-AdminFolderUid <Int32>]
   [-ApplicationGroupName <String>]
   [-AssociatedDesktopGroupPriority <Int32>]
   [-AssociatedDesktopGroupUid <Int32>]
```



```
    [-AssociatedDesktopGroupUUID <Guid>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserSID <String>]
    [-AssociatedUserUPN <String>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-Metadata <String>]
    [-RestrictToTag <String>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-SessionSharingEnabled <Boolean>]
    [-SingleAppPerSession <Boolean>]
    [-Tag <String>]
    [-TenantId <Guid>]
    [-TotalApplications <Int32>]
    [-TotalMachines <Int32>]
    [-TotalMachinesWithTagRestriction <Int32>]
    [-UserFilterEnabled <Boolean>]
    [-UUID <Guid>]
    [-ApplicationUid <Int32>]
    [-DesktopGroupUid <Int32>]
    [-TagUid <Int32>]
    [-UserSID <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerApplicationGroup
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerApplicationGroup cmdlet returns application groups that have been configured as part of the site.

With no parameters, Get-BrokerApplicationGroup returns all the application groups that have been configured. You also can use the parameters of Get-BrokerApplicationGroup to filter the results to just application groups you're interested in. You can also identify application groups by their UIDs or their Names.

See about_Broker_Applications for more information.

——————————BrokerApplicationGroup Object

A BrokerApplicationGroup object represents a single application group that has been configured as part of the site. It has the following properties:

- AdminFolderName (System.String)

  The name of the admin folder the application group is in (including trailing backslash), or the empty string if the application group is at the root level

- AdminFolderUid (System.Int32)

  The Uid of the admin folder the application group is in (if any)

- ApplicationGroupName (System.String)

  Name of the application group (must be unique within a Folder)

- AssociatedDesktopGroupPriorities (System.Int32[])

  List of associated desktop group priorities. This list is presented in the same order as AssociatedDesktopGroupUids and AssociatedDesktopGroupUUIDs.
  When launching an application an available machine from one of the associated desktop groups is selected. Desktop groups are searched for available machines in order of their priority.

- AssociatedDesktopGroupUids (System.Int32[])

  List of associated desktop group uids. Associated desktop groups is the list of desktop groups on which the application group is published. The list is sorted by priority, with the highest priority desktop group first.

- AssociatedDesktopGroupUUIDs (System.Guid[])

  List of associated desktop group UUIDs. Associated desktop groups is the list of desktop groups on which the application group is published. The list is sorted by priority, with the highest priority desktop group first.

- AssociatedUserFullNames (System.String[])

  List of associated users (full names). Associated users is the list of users who are given access using the application group/user mapping filter.

- AssociatedUserNames (System.String[])

  List of associated users (SAM names). Associated users is the list of users who are given access using the application group/user mapping filter.

- AssociatedUserSIDs (System.String[])

  List of associated users (SIDs). Associated users is the list of users who are given access using the application group/user mapping filter.

- AssociatedUserUPNs (System.String[])

  List of associated users (user principle names). Associated users is the list of users who are given access using the application group/user mapping filter.

- Description (System.String)

  Optional application group description. This description is visible in Citrix Studio and can be used for any purpose. As with other facets of application groups, the description is not visible to end users.

- Enabled (System.Boolean)

  Specifies whether or not the applications in this application group can be launched.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this application group.

- Name (System.String)

  Site-wide unique full path name of this application group including all containing folders, for example, Folder1\Folder2\MyApplicationGroup The name is visible in Citrix Studio and uniquely identifies the application group. As with other facets of application groups, the name is not visible to end users.

- RestrictToTag (System.String)

  Optional tag that may be used further to restrict which machines may be used for launching the application group's applications. A machine may be used by an application group if either the application group has no tag restriction or the application group does have a tag restriction and the machine is tagged with the same tag.

- Scopes (Citrix.Fma.Sdk.DelegatedAdminInterfaces.ScopeReference[])

  The list of the delegated admin scopes to which the application group belongs.

- SessionSharingEnabled (System.Boolean)

  Whether applications in this application group can share sessions with applications in other application groups (or with applications that are not a member of an application group). Applications in the same application group can always share sessions with each other, unless session sharing has been disabled by other means.

- SingleAppPerSession (System.Boolean)

  Indicates whether each application launched from this application group uses a new session or can share an existing one.

- Tags (System.String[])

  Tags associated with the application group.

- TenantId (System.Guid?)

  Identity of tenant associated with application group. Not applicable (always blank) in non-multitenant sites.

- TotalApplications (System.Int32)

  Number of applications in the application group.

- TotalMachines (System.Int32)

  Total number of machines across all desktop groups on which the application group is published.

- TotalMachinesWithTagRestriction (System.Int32)

  Total number of machines across all desktop groups on which the application group is published, and which are tagged with the tag given by the RestrictToTag property.
  The value of this property is always equal to or less than the value of the TotalMachines property. If RestrictToTag is $null, then the properties' values are equal.

- Uid (System.Int32)

  Uid of the application group.

- UserFilterEnabled (System.Boolean)

  Whether the application group's user filter is enabled.

- UUID (System.Guid)

  UUID of the application group.

Examples

EXAMPLE 1

Gets application groups whose name starts with "Account".

```
1 Get-BrokerApplicationGroup -Name "Account*"
```

EXAMPLE 2

Gets application groups for which session sharing with applications in other application groups has been disabled.

```
1 Get-BrokerApplicationGroup -SessionSharingEnabled $false
```


Parameters

-Uid

Gets the application group with the given Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets application groups whose full path name matches the supplied pattern.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderName

Gets application groups that are in admin folders matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-AdminFolderUid

Gets application groups that are in the specified admin folder.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationGroupName

Gets application groups that match the specified simple name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedDesktopGroupPriority

Gets application groups with which a desktop group has been associated with the specified priority.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedDesktopGroupUid

Gets application groups which have been associated with the specified desktop group. The desktop group is identified by its Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedDesktopGroupUUID

Gets application groups which have been associated with the specified desktop group. The desktop group is identified by its UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserFullName

Gets application groups with an associated user identified by their user name (usually 'first-name last-name').

If the 'UserFilterEnabled'property is true then access to applications in the application group is restricted to those users only. Otherwise, access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets application groups with an associated user identified by their user name (in the form 'domain\user').

If the 'UserFilterEnabled'property is true then access to applications in the application group is restricted to those users only. Otherwise, access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserSID

Gets application groups with an associated user identified by their Windows SID.

If the 'UserFilterEnabled'property is true then access to applications in the application group is restricted to those users only. Otherwise, access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserUPN

Gets application groups with an associated user identified by their user principle name (in the form 'user@domain').

If the 'UserFilterEnabled'property is true then access to applications in the application group is restricted to those users only. Otherwise, access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets application groups whose description matches the supplied pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Enabled

Gets application groups which are currently enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

Gets only application groups with the specified tag restriction.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-ScopeId

Gets application groups which are part of the supplied administrative scope. The scope is identified by its GUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets application groups which are part of an administrative scope whose name matches the supplied pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionSharingEnabled

Gets application groups for which session sharing is enabled.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SingleAppPerSession

Specifies whether each application launched from this application group starts in its own new session or can share an existing suitable session if present.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets application groups that have been tagged with a tag whose name matches the specified pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantId

Gets desktop groups associated with the specified tenant identity.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalApplications

Gets application groups that contain the specified number of applications.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalMachines

Gets application groups that are published on the specified number of machines, without taking the tag restriction into account.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalMachinesWithTagRestriction

Gets application groups that are published on the specified number of machines, taking the tag restriction into account.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Gets application groups whose user filter is currently enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets the application group with the given UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets only application groups to which the given application has been added.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets application groups which have been added to the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TagUid

Gets application groups that have been tagged the given tag. The tag is identified by its Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserSID

Gets application groups for which the specified user is a member of the user filter. The user account is identified by its SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ApplicationGroup

Get-BrokerApplicationGroup returns an object for each matching application group.

Notes

Application groups are explained in about_Broker_Applications.

To perform greater-than or less-than comparisons, use -Filter. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Applications
- Add-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Rename-BrokerApplicationGroup
- Move-BrokerApplicationGroup
- Set-BrokerApplicationGroup

Get-BrokerApplicationInstance

March 11, 2024

Gets the running applications on the desktops.

Syntax

```
Get-BrokerApplicationInstance
   [-ApplicationName <String>]
   [-ApplicationUid <Int32>]
   [-ApplicationUUID <Guid>]
   [-Instances <Int32>]
   [-MachineName <String>]
   [-MachineUid <Int32>]
   [-Metadata <String>]
   [-SessionKey <Guid>]
   [-SessionUid <Int64>]
   [-UserName <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerApplicationInstance
   -Uid <Int64>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Get-BrokerApplicationInstance gets the published applications that are running on desktops.

Only published applications that are launched from a Citrix client are returned. If a user launches an application from within a session (by double-clicking on an attachment from an email, for example) this will not show up in the list of running applications.

Also note that this is a list of launched published applications, not a list of processes running on the desktop. In some cases the original process associated with the published application may no longer be running, but if the session is still running the published application may be listed as running.

The number of instances for each published application running in a session is also returned. For example, if a user launches two Notepad applications from a Citrix client, and session-sharing occurs such that both Notepad applications run in the same session, then the Instances property indicates that 2 copies are running in the session.

See Get-BrokerApplication and Get-BrokerSession to get the details for the applications and sessions, respectively.

The Get-BrokerMachine cmdlet also returns a list of published applications that are running on a desktop. See the "ApplicationsInUse"attribute of the returned desktop objects.

——————————BrokerApplicationInstance Object

The BrokerApplicationInstance object represents an instance of a published application in the site. It contains the following properties:

- ApplicationName (System.String)

  The administrative name of the application.

- ApplicationUid (System.Int32)

  The UID of the application.

- ApplicationUUID (System.Guid)

  The UUID of the application.

- Instances (System.Int32)

  The number of times this published application is running in the specified session.

- MachineName (System.String)

  Machine's SAM name (of the form domain\machine). If SAM name is unavailable, contains the machines's SID.

- MachineUid (System.Int32)

  UID of underlying machine.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this application instance.

- SessionKey (System.Guid)

  The key of the session.

- SessionUid (System.Int64)

  The UID of the session.

- Uid (System.Int64)

  The unique identifier for this application instance object itself, distinct from the Uids of either application or session. objects.

- UserName (System.String)

  User name (SAMName).

Examples

EXAMPLE 1

Returns the application instance with a Uid of 3. Note that this
is the unique identifier for application instances, which is distinct from the
unique identifiers of either application or session objects.

```
Get-BrokerApplicationInstance -Uid 3
```

EXAMPLE 2

Returns all the application instances for the Notepad application.
Use this to see if there are any launched instances of Notepad
running in your site and, if so, from which desktops.

```
$app = Get-BrokerApplication Notepad
Get-BrokerApplicationInstance -ApplicationUid $app.Uid
```

EXAMPLE 3

Returns all the applications that are running on the “Worker1”machine
in the “ACME”domain. Use this to see which published applications
are running on a specific machine.
Note that the SessionUid, not the SessionId, is specified as a parameter to this
cmdlet. The SessionId is a unique identifier that Remote Desktop Services uses to
track the session, and is unique only on that machine. The SessionUid, on the other
hand, is unique across the entire site.
The “ApplicationsInUse”attribute of the returned session object also provides

a list of running launched applications, and in many cases might be more convenient to use. It returns a list of application BrowserNames.

```
$sessions = Get-BrokerSession -MachineName "ACME\Worker1"
for ($i=0; $i -lt $sessions.Length; $i++) {

            Get-BrokerApplicationInstance -SessionUid $sessions[$i].
               SessionUid
           }
```

Parameters

-Uid

Gets only the application instances specified by the unique identifier. This is the unique identifier for the application instance object itself, and is distinct from the Uids of either application or session objects.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationName

Gets only application instances for the specified application name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-ApplicationUid

Gets only application instances for the specified application Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUUID

Gets only the application instances for the specified application UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Instances

Gets only application instances that match the specified number of instances.

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets only application instances running on the specified machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineUid

Gets only application instances running on the machine with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionKey

Gets only application instances for the published applications running in the specified session.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets only application instances for the published applications running in the specified session.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserName

Gets only application instances being run by the specified users.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ApplicationInstance

Get-BrokerApplicationInstance returns an object for each application instance it gets.

Related Links

- about_Broker_Applications
- Get-BrokerApplication
- Get-BrokerDesktop
- Get-BrokerSession

Get-BrokerAssignmentPolicyRule

March 11, 2024

Gets desktop rules from the site's assignment policy.

Syntax

```
Get-BrokerAssignmentPolicyRule
   [[-Name] <String>]
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   [-DesktopGroupUid <Int32>]
   [-Enabled <Boolean>]
```



```
    [-ExcludedUser <User>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-IconUid <Int32>]
    [-IncludedUser <User>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-MaxDesktops <Int32>]
    [-Metadata <String>]
    [-PublishedName <String>]
    [-SecureIcaRequired <Boolean>]
    [-UUID <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerAssignmentPolicyRule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns desktop rules matching the specified search criteria from the site's assignment policy. If no search criteria are specified, all desktop rules in the assignment policy are obtained.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

——————————BrokerAssignmentPolicyRule Object

The BrokerAssignmentPolicyRule object represents a single desktop rule within the site's assignment policy. It contains the following properties:

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

  The color depth of desktop sessions launched by the user from machines assigned to them by the rule. If null, the equivalent setting from the rule's desktop group is used.

- Description (System.String)

  Optional description of the rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

- DesktopGroupUid (System.Int32)

  The unique ID of the desktop group to which the rule applies.

- Enabled (System.Boolean)

  Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's assignment policy.

- ExcludedUserFilterEnabled (System.Boolean)

  Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

- ExcludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to machine assignments from the rule's desktop group.

- IconUid (System.Int32?)

  The unique ID of the icon used for the machine entitlement seen by the user or, after a machine is assigned by the rule, the icon for the desktop itself. If null, the equivalent setting from the rule's desktop group is used.

- IncludedUserFilterEnabled (System.Boolean)

  Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly entitled to the machine assignments described by the rule.
  For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The included users filter of the rule, that is, the users and groups who are entitled to machine assignments from the rule's desktop group.

- MaxDesktops (System.Int32)

  The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.

- Name (System.String)

The administrative name of the rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

- PublishedName (System.String)

  The published name of the desktop entitlement seen by the user or, after a machine is assigned by the rule, the published name of the desktop itself. If null, the equivalent setting from the rule's desktop group is used.

- SecureIcaRequired (System.Boolean?)

  Indicates whether the rule requires the SecureICA protocol for desktop sessions launched using a machine assigned by the rule. If null, the equivalent setting from the rule's desktop group is used.

- Uid (System.Int32)

  The unique ID of the rule itself.

- UUID (System.Guid)

  UUID of the rule.

Examples

EXAMPLE 1

Returns all desktop rules from the assignment policy. This offers a complete description of the current site's assignment policy with respect to machine assignment entitlements for delivery of desktop sessions from private desktop groups.

```
Get-BrokerAssignmentPolicyRule
```

EXAMPLE 2

Returns all rules in the assignment policy that give users entitlements to machine assignments in the Sales Support desktop group for delivery of full desktop sessions.

```
$dg = Get-BrokerDesktopGroup 'Sales Support'
Get-BrokerAssignmentPolicyRule -DesktopGroupUid $dg.Uid
```

Parameters

-Uid

Gets the desktop rule with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only desktop rules with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ColorDepth

Gets only desktop rules with the specified color depth.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets only desktop rules with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only desktop rules that apply to the desktop group with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only rules that are in the specified state, either enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUser

Gets only desktop rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Gets only desktop rules that have their excluded user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets only desktop rules using the icon with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUser

Gets only desktop rules that have the specified user in their included users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Gets only desktop rules that have their included user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxDesktops

Gets only desktop rules granting the specified number of machine assignment entitlements.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Gets only desktop rules with the specified published name, that is, the desktop name that the end user sees.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SecureIcaRequired

Gets only desktop rules that require desktop sessions to machines assigned by the rule to use the SecureICA protocol ($true) or not ($false).

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets rules with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

Get-BrokerAssignmentPolicyRule returns all assignment policy rules that match the specified selection criteria.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_ErrorHandling
- New-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule
- Remove-BrokerAssignmentPolicyRule

Get-BrokerAutoTagRule

March 11, 2024

Gets AutoTagRules that have been defined for this site.

Syntax

```
Get-BrokerAutoTagRule
    [[-Name] <String>]
    [-Description <String>]
    [-Metadata <String>]
    [-ObjectType <String>]
    [-RuleText <String>]
    [-TagUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerAutoTagRule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves AutoTagRules matching the specified criteria. If no parameters are specified this cmdlet enumerates all catalogs.

See about_Broker_Filtering for information about advanced filtering options.

————————BrokerAutoTagRule Object

The BrokerAutoTagRule object represents a single instance of an AutoTagRule.

See about_Broker_AutoTagRule for more information.

A BrokerAutoTagRule contains the following properties:

- Description (System.String)

Description of the AutoTagRule

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  The metadata for the AutoTagRule.

- Name (System.String)

  The name of the AutoTagRule

- ObjectType (System.String)

  The object type the rule is applied to.

- RuleText (System.String)

  The rule in a text format

- TagUid (System.Int32)

  The Tag Uid this rule sets or removes

- Uid (System.Int32)

  The Uid of the AutoTagRule

Examples

EXAMPLE 1

Gets the AutoTagRules with the name RandomAllocatedCatalogs

```
1  Get-AutoTagRule -Name RandomAllocatedCatalogs
```

Parameters

-Uid

Gets the AutoTagRule with the unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Name

Gets AutoTagRules that have the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets the AutoTagRules that have the specified Description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ObjectType

Gets the AutoTagRules that operate on the ObjectType.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RuleText

Gets the AutoTagRules that have the specified RuleText.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TagUid

Gets the AutoTagRule associated with the TagUid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AutoTagRule

Get-AutoTagRule returns matching AutoTagRule objects.

Related Links

- about_Broker_AutoTagRule
- New-BrokerAutoTagRule
- Set-BrokerAutoTagRule
- Remove-BrokerAutoTagRule
- Rename-BrokerAutoTagRule

Get-BrokerCatalog

March 11, 2024

Gets catalogs configured for this site.

Syntax

```
Get-BrokerCatalog
   [[-Name] <String>]
   [-AdminFolderName <String>]
   [-AdminFolderUid <Int32>]
```



```
    [-AllocationType <AllocationType>]
    [-AppDnaAnalysisState <AppDnaAnalysisState>]
    [-AssignedCount <Int32>]
    [-AvailableAssignedCount <Int32>]
    [-AvailableCount <Int32>]
    [-AvailableUnassignedCount <Int32>]
    [-CatalogName <String>]
    [-Description <String>]
    [-HypervisorConnectionUid <Int32>]
    [-IsRemotePC <Boolean>]
    [-MachinesArePhysical <Boolean>]
    [-MdmEnrollment <String>]
    [-Metadata <String>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-PersistUserChanges <PersistUserChanges>]
    [-ProvisioningSchemeId <Guid>]
    [-ProvisioningType <ProvisioningType>]
    [-PvsAddress <String>]
    [-PvsDomain <String>]
    [-RemotePCDesktopGroupPriority <Int32>]
    [-RemotePCDesktopGroupUid <Int32>]
    [-RemotePCHypervisorConnectionUid <Int32>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-SessionSupport <SessionSupport>]
    [-Tag <String>]
    [-TenantId <Guid>]
    [-TimeZone <String>]
    [-UnassignedCount <Int32>]
    [-UsedCount <Int32>]
    [-UUID <Guid>]
    [-ZoneHealthy <Boolean>]
    [-ZoneName <String>]
    [-ZoneUid <Guid>]
    [-MachineUid <Int32>]
    [-TagUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerCatalog
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Retrieves catalogs matching the specified criteria. If no parameters are specified this cmdlet enumerates all catalogs.

See about_Broker_Filtering for information about advanced filtering options.

——————————BrokerCatalog Object

The catalog object returned represents a group of related physical or virtual machines that have been configured in the site.

See about_Broker_Machines for more information.

- AdminFolderName (System.String)

  The name of the admin folder the catalog is in (including trailing backslash), or the empty string if the catalog is at the root level

- AdminFolderUid (System.Int32)

  The Uid of the admin folder the catalog is in (if any)

- AllocationType (Citrix.Broker.Admin.SDK.AllocationType)

  Denotes how the the machines in the catalog are allocated to a user.
  Possible values are:

  - Static: Machines get assigned to a user either by the admin or on first use. This relationship is static and changes only if an admin explicitly changes the assignments.
  - Random: Machines are allocated to users randomly from a pool of available machines.

- AppDnaAnalysisState (Citrix.Broker.Admin.SDK.AppDnaAnalysisState?)

  The AppDNA Analysis State of the catalog

- AssignedCount (System.Int32)

  The number of assigned machines (machines that have been assigned to a user/users or a client name/address).

- AvailableAssignedCount (System.Int32)

  The number of available machines (not in a desktop group), that are also assigned to users.

- AvailableCount (System.Int32)

  The number of available machines (those not in any desktop group).

- AvailableUnassignedCount (System.Int32)

  The number of available machines (those not in any desktop group) that are not assigned to users.

- CatalogName (System.String)

  Name of the catalog (must be unique within a Folder)

- Description (System.String)

  Description of the catalog.

- HypervisorConnectionUid (System.Int32?)

  The Uid of the hypervisor connection that is associated with the machines in the catalog. This property only applies to MCS provisioned catalogs. For other provisioning types machines can be from one or more different hypervisor connections.

- IsRemotePC (System.Boolean)

  Specifies whether or not the catalog is a RemotePC catalog. Remote PC catalogs automatically configure appropriate machines without the need for manual configuration. See about_Broker_RemotePC for more information.

- MachinesArePhysical (System.Boolean)

  Specifies whether or not the machines in the catalog can be power-managed by the broker.

- MdmEnrollment (System.String)

  Indicates the MdmEnrollment of the catalog.
  Possible values:

  - None: No Mdm.
  - Intune: Microsoft Intune.
  - IntuneWithCitrixTags: Microsoft Intune with Citrix tags.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Holds any metadata associated with the catalog.

- MinimumFunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel)

  The expected minimal functional level of the machines in the catalog.

- Name (System.String)

  Site-wide unique full path name of the catalog including all containing folders, for example, Folder1\Folder2\MyCatalog.

- PersistUserChanges (Citrix.Broker.Admin.SDK.PersistUserChanges)

  Specifies how user changes are persisted on machines in the catalog. Possible values are:

  - OnLocal: User changes are stored on the machine's local storage.
  - Discard: User changes are discarded.

- ProvisioningSchemeId (System.Guid?)

  The GUID of the provisioning scheme (if any) associated with the catalog. This only applies if the provisioning type is MCS.

- ProvisioningType (Citrix.Broker.Admin.SDK.ProvisioningType)

  Specifies how the machines are provisioned in the catalog. Possible values are:

  - Manual: No provisioning.
  - PVS: Machine provisioned by PVS (machine may be physical, blade, VM…)
  - MCS: Machine provisioned by MCS (machine must be a VM).

- PvsAddress (System.String)

  IP address of the PVS server to be used in a catalog with a PVS ProvisioningType.

- PvsDomain (System.String)

  The domain of the PVS server to be used in a catalog with a PVS ProvisioningType.

- RemotePCDesktopGroupPriorities (System.Int32[])

  Remote PC desktop groups'association priorities.

- RemotePCDesktopGroupUids (System.Int32[])

  UIDs of the Remote PC desktop groups associated with this catalog.

- RemotePCHypervisorConnectionUid (System.Int32?)

  UID of the hypervisor connection used for powering on RemotePC machines in this catalog (only for catalogs with IsRemotePC set to true).

- Scopes (Citrix.Fma.Sdk.DelegatedAdminInterfaces.ScopeReference[])

  The list of the delegated admin scopes to which the catalog belongs.

- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport)

  Specifies the session support of the machines in the catalog. Valid values are: SingleSession, MultiSession.

- Tags (System.String[])

  Tags associated with the catalog.

- TenantId (System.Guid?)

  Identity of tenant associated with catalog. Not applicable (always blank) in non-multitenant sites.

- TimeZone (System.String)

  The timezone that desktops in the catalog are in (for catalog reboot schedule purposes).

- Uid (System.Int32)

  Uid of the catalog.

- UnassignedCount (System.Int32)

  The number of unassigned machines (machines not assigned to users).

- UsedCount (System.Int32)

  The number of machines in the catalog that are in a desktop group.

- UUID (System.Guid)

  The global ID of the catalog.

- ZoneHealthy (System.Boolean?)

  Health state of the Zone associated with this catalog.

- ZoneName (System.String)

  Name of the Zone associated with this catalog.

- ZoneUid (System.Guid)

  Uid of the Zone associated with this catalog.

Examples

EXAMPLE 1

These commands return all catalogs containing machines that are randomly assigned to users. The second form uses advanced filtering (see about_Broker_Filtering).

```
Get-BrokerCatalog -AllocationType Random
Get-BrokerCatalog -Filter {
  AllocationType  -eq 'Random' }
```

EXAMPLE 2

This command returns catalogs with more than 10 unused machines that are available for assignment to users.

```
Get-BrokerCatalog -Filter {
  AvailableCount  -gt 10 }
```


EXAMPLE 3

This command returns the unmanaged catalog with the highest number of available machines.

```
Get-BrokerCatalog -MaxRecordCount 1 -ProvisioningType Manual -SortBy '-
    AvailableCount'
```

Parameters

-Uid

Get catalogs with the specified UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets catalogs whose full path name matches the specified supplied pattern.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderName

Gets catalogs that are in admin folders matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderUid

Gets catalogs that are in the specified admin folder.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllocationType

Gets catalogs that are of the specified allocation type. Values can be:

- Static - Machines in a catalog of this type are permanently assigned to a user.
- Permanent - equivalent to 'Static'.
- Random - Machines in a catalog of this type are picked at random and temporarily assigned to a user.

Type:	AllocationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AppDnaAnalysisState

Gets catalogs that have of the specified AppDNA Analysis State. Values can be:

- None - No AppDNA analysis has not been preformed.
- Capturing - AppDNA analysis is running and capturing infomation from a catalog machine.
- Canceled - The AppDNA analysis process was cancelled.
- Ready - The AppDNA analysis process has finished correctly.
- Failed - The AppDNA analysis process failed.
- Importing - AppDNA analysis is running and uploading results to the AppDNA server.

Type:	AppDnaAnalysisState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedCount

Gets catalogs containing a specified number of assigned machines (machines that have been assigned to users).

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AvailableAssignedCount

Gets catalogs containing a specified number of available machines (those not in any desktop group) that are also assigned to users.

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AvailableCount

Gets catalogs containing a specified number of available machines (those not in any desktop group).

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AvailableUnassignedCount

Gets catalogs containing a specified number of available machines (those not in any desktop group) that are not assigned to users.

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets Catalogs that match the specified simple name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets catalogs with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets catalogs associated with the specified hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsRemotePC

Gets catalogs with the specified IsRemotePC value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesArePhysical

Specifies whether machines in the catalog can be power-managed by the Citrix Broker Service. Where the Citrix Broker Service cannot control the power state of the machine specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the legacy CatalogKind parameter only with a Pvs catalog kind.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MdmEnrollment

Gets catalogs that have the specified MdmEnrollment. Values can be:

- None - No enrollment.
- Intune - Microsoft Intune.

Type:	String
Accepted values:	Intune, IntuneWithCitrixTags, None
Position:	Named
Default value:	If no value is provided, the catalog does not enroll with Mdm.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MinimumFunctionalLevel

Gets catalogs with a specific MinimumFunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25, L7_30, L7_34

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PersistUserChanges

Gets catalogs with the specified behavior when persisting changes made by the end user. Possible values are:

- OnLocal - User changes are stored on the machine's local storage.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeId

Gets catalogs associated with the specified provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningType

Specifies the provisioning type for the catalog. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,…).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvsAddress

Gets catalogs containing machines provided by the Provisioning Services server with the specified address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PvsDomain

Gets catalogs containing machines provided by the Provisioning Services server in the specified domain.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RemotePCDesktopGroupPriority

Gets Remote PC catalogs with a Remote PC desktop group association with the specified priority.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemotePCDesktopGroupUid

Gets Remote PC catalogs associated with the specified Remote PC desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemotePCHypervisorConnectionUid

Gets Remote PC catalogs associated with the specified Remote PC hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets catalogs that are associated with the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets catalogs that are associated with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionSupport

Gets catalogs that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets catalogs tagged with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantId

Gets catalogs associated with the specified tenant identity.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TimeZone

Gets catalogs with the specified value of TimeZone.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UnassignedCount

Gets catalogs containing a specified number of unassigned machines (machines not assigned to users).

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UsedCount

Gets catalogs containing a specified number of machines used in a desktop group.

This property is typically used with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Get catalogs with the specified global ID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneHealthy

Gets catalogs located in the zone with the specified health state.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets catalogs located in the zone with the specified name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets machines located in the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets the catalog containing the machine referenced by the specified unique identifier (UID).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TagUid

Gets catalogs to which the specified tag (identified by its Uid) has been added to help identify it - see Add-BrokerTag for more information.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Catalog

Get-BrokerCatalog returns an object for each matching catalog.

Related Links

- about_Broker_Machines
- about_Broker_RemotePC
- about_Broker_Filtering
- New-BrokerCatalog
- Remove-BrokerCatalog
- Rename-BrokerCatalog
- Set-BrokerCatalog
- Add-BrokerTag
- Move-BrokerCatalog

Get-BrokerCatalogRebootSchedule

March 11, 2024

Gets one or more catalog reboot schedules.

Syntax

```
Get-BrokerCatalogRebootSchedule
    [[-Name] <String>]
    [-Active <Boolean>]
    [-CatalogName <String>]
    [-CatalogUid <Int32>]
    [-Enabled <Boolean>]
    [-MaxOvertimeStartMins <Int32>]
    [-RebootDuration <Int32>]
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-WarningRepeatInterval <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerCatalogRebootSchedule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerCatalogRebootSchedule cmdlet is used to enumerate catalog reboot schedules that match all of the supplied criteria.

A catalog reboot schedule can be configured to cause all of the machines in a catalog to be rebooted once at a particular time and day, with the reboot of the individual machines spread out over the duration of the whole reboot cycle. A specific warning message can be configured to be displayed to users who are running sessions on the machines being rebooted.

See about_Broker_Filtering for information about advanced filtering options.

——————————BrokerCatalogRebootSchedule Object

The catalog reboot schedule object returned represents a scheduled reboot of machines in a catalog.

- Active (System.Boolean)

  True if there is an active reboot cycle for this schedule, false otherwise.

- CatalogName (System.String)

  Name of the catalog rebooted by this schedule.

- CatalogUid (System.Int32)

  Uid of the catalog rebooted by this schedule.

- Description (System.String)

  An optional description for the reboot schedule.

- Enabled (System.Boolean)

  True if this schedule is currently enabled, false otherwise.

- MaxOvertimeStartMins (System.Int32)

  Maximum delay in minutes after which the scheduled reboot will not take place

- Name (System.String)

  Name of the reboot schedule.

- RebootDuration (System.Int32)

  Approximate maximum number of minutes over which the scheduled reboot cycle runs.

- StartDate (System.String)

  The date on which the schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD).

- StartTime (System.TimeSpan)

  Time of day at which the scheduled reboot cycle starts.

- Uid (System.Int32)

  Uid of the reboot schedule.

- WarningDuration (System.Int32)

  Number of minutes to display the warning message for.

- WarningMessage (System.String)

  Warning message to display to users in active sessions prior to rebooting the machine.

- WarningRepeatInterval (System.Int32)

  Number of minutes to wait before displaying the warning message again.

- WarningTitle (System.String)

  Title of the warning message dialog.

Examples

EXAMPLE 1

Enumerates all of the catalog reboot schedules.

```
Get-BrokerCatalogRebootSchedule
```

EXAMPLE 2

Enumerates all disabled catalog reboot schedules.

```
Get-BrokerCatalogRebootSchedule -Enabled $false
```

EXAMPLE 3

Returns the catalog reboot schedules for the catalog having the Uid 11.

```
Get-BrokerCatalogRebootSchedule -CatalogUid 11
```

Parameters

-Uid

Gets the reboot schedule with the specified Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets the reboot schedule with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Active

Gets desktop group reboot schedules according to whether they are currently active or not. A schedule is active if there is a reboot cycle currently running that was started as a result of the schedule.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets the reboot schedules for the catalog having this name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogUid

Gets the reboot schedules for the catalog having this Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets the reboot schedules with the specified Enabled value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Gets the reboot schedules with the specified duration.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDate

Gets the reboot schedules with the specified start date (YYY-MM-DD).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-StartTime

Gets the reboot schedules with the specified start time (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Gets the reboot schedules with the specified warning repeat interval.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.CatalogRebootSchedule

Returns matching catalog reboot schedules.

Related Links

- Set-BrokerCatalogRebootSchedule
- New-BrokerCatalogRebootSchedule
- Remove-BrokerCatalogRebootSchedule
- Rename-BrokerCatalogRebootSchedule
- Get-BrokerRebootCycle
- about_Broker_Filtering

Get-BrokerConfigurationSlot

March 11, 2024

Gets configuration slots configured for this site.

Syntax


```
Get-BrokerConfigurationSlot
    [[-Name] <String>]
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerConfigurationSlot
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Get the list of configuration slots defined for this site. Each configuration slot determines a collection of related settings that can be specified in a machine configuration associated with this slot.

For example, a configuration slot may be defined to configure only "User Profile Manager"settings.

See about_Broker_Filtering for information about advanced filtering options.

————————BrokerConfigurationSlot Object

The configuration slot object returned represents a named collection of related settings.

- Description (System.String)

  Optional description of this configuration slot.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  A map of metadata associated with this configuration slot.

- Name (System.String)

  Unique name of this configuration slot.

- SettingsGroup (System.String)

  The encoded identity of the settings group that every setting in the associated machine configuration instances must belong to.

- Uid (System.Int32)

  Unique Uid of this configuration slot.

Examples

EXAMPLE 1

Retrieves every configuration slot.

```
Get-BrokerConfigurationSlot
```

EXAMPLE 2

Retrieves the configuration slot named “AppV”.

```
Get-BrokerConfigurationSlot -Name "AppV"
```

Parameters

-Uid

Get only the configuration slot with the specified unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Get only the configuration slot with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ConfigurationSlot

Get-BrokerConfigurationSlot returns an object for each matching slot.

Related Links

- New-BrokerConfigurationSlot
- Remove-BrokerConfigurationSlot
- Get-BrokerMachineConfiguration
- about_Broker_Filtering
- about_Broker_ConfigurationSlots

Get-BrokerConfiguredFTA

March 11, 2024

Gets any file type associations configured for an application.

Syntax

```
Get-BrokerConfiguredFTA
    [-ApplicationUid <Int32>]
    [-ContentType <String>]
    [-ExtensionName <String>]
    [-HandlerDescription <String>]
    [-HandlerName <String>]
    [-HandlerOpenArguments <String>]
    [-UUID <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerConfiguredFTA
   -Uid <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets any file type associations that are configured for content redirection to a published application.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Configured file type associations are different from imported file type associations. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection. Imported file type associations are lists of known file type associations for a given desktop group. See Update-BrokerImportedFTA for more information about imported file type associations.

——————————BrokerConfiguredFTA Object

The BrokerConfiguredFTA object represents a file type association configured for a published application. It contains the following properties:

- ApplicationUid (System.Int32)

The Uid of the application configured for the file type association.

- ContentType (System.String)

  Content type of the file, such as “text/plain”or “application/vnd.ms-excel”.

- ExtensionName (System.String)

  A single file extension, such as .txt, unique within the scope of a desktop group.

- HandlerDescription (System.String)

  File type description, such as “Test Document”, “Microsoft Word Text Document”, etc.

- HandlerName (System.String)

  File type handler name, e.g. “Word.Document.8”or TXTFILE.

- HandlerOpenArguments (System.String)

  The arguments used for the ‘open’action on files of this type.

- Uid (System.Int32)

  Unique internal identifier of configured file type association.

- UUID (System.Guid)

  UUID of the configured file type association.

Examples

EXAMPLE 1

Returns all configured file type associations.

```
Get-BrokerConfiguredFTA
```

EXAMPLE 2

Returns only configured file type associations

that have a “.txt”extension.

```
Get-BrokerConfiguredFTA -ExtensionName ".txt"
```


Parameters

-Uid

Gets only the configured file type association for the specified unique identifier.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets only the configured file type associations for the specified application unique identifier.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ContentType

Gets only the configured file type associations for the specified content type (as seen in the Registry). For example, “text/plain”or “application/msword”.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExtensionName

Gets only the configured file type associations for the specified extension name. For example, “.txt”or “.doc”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HandlerDescription

Gets only the configured file type associations for the specified handler description. For example, “Text Document”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HandlerName

Gets only the configured file type associations for the

specified handler name. For example, “TXTFILE”or “Word.Document.8”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HandlerOpenArguments

Gets only the configured file type associations for the

specified open argument to the handler. For example, “%1”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UUID

Gets configured file type associations with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32

Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

No input is accepted from the pipeline.

Outputs

Citrix.Broker.Admin.SDK.ConfiguredFTA

This cmdlet returns one or more ConfiguredFTA objects.

Related Links

- New-BrokerConfiguredFTA
- Remove-BrokerConfiguredFTA
- Update-BrokerImportedFTA

Get-BrokerConnectionLog

March 11, 2024

Get entries from the site's session connection log.

Syntax

```
Get-BrokerConnectionLog
   [[-MachineName] <String>]
   [-BrokeringTime <DateTime>]
   [-BrokeringUserName <String>]
   [-BrokeringUserUPN <String>]
   [-ConnectionFailureReason <ConnectionFailureReason>]
   [-Disconnected <Boolean>]
   [-EndTime <DateTime>]
   [-EstablishmentTime <DateTime>]
   [-MachineDNSName <String>]
   [-MachineUid <Int32>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
```



```
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerConnectionLog
    [-Uid] <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets connection log entries matching the specified criteria. If no parameters are specified all connection log entries are returned.

The session connection log contains entries describing each brokered connection, or reconnection, attempt to a session in the site.

Each log entry describes a single connection brokering attempt to a new or existing session within the site. A single session can have multiple entries in the connection log, for example where the end user brokers a connection to a new session, disconnects and later brokers a reconnection. Conversely, other sessions may have none (e.g. console sessions).

By default connection log entries are removed after 48 hours.

For information about advanced filtering options when using the -Filter parameter, see about_Broker_Filtering; for information about machines, see about_Broker_Machines.

————————BrokerConnectionLog Object

The BrokerConnectionLog object represents a single brokered connection attempt to a new or existing session on a machine in the site. It contains the following properties:

- BrokeringTime (System.DateTime)

  The time at which the connection attempt was made.

- BrokeringUserName (System.String)

  The name of the user making the connection (in DOMAIN\User format).

- BrokeringUserUPN (System.String)

  The name of the user making the connection (in `user@upndomain.com` format).

- ConnectionFailureReason (Citrix.Broker.Admin.SDK.ConnectionFailureReason?)

  The status of the connection attempt. A value of None indicates that the connection was successfully established, $null that the attempt is still in progress, and other values indicate that the attempt failed for the specified reason.

- Disconnected (System.Boolean?)

  Indicates if the connection was ended by disconnection (True), logoff or establishment failure (False), or is still active ($null).

- EndTime (System.DateTime?)

  The time at which the connection ended. If the connection ended by disconnection, the underlying machine session would still exist in a disconnected state.

- EstablishmentTime (System.DateTime?)

  The time at which the connection was successfully established. The value is $null if the connection attempt failed or is still in progress.

- MachineDNSName (System.String)

  The name of the machine to which the connection was made (in `machine@dnsdomain.com` form).

- MachineName (System.String)

  The name of the machine to which the connection was made (in DOMAIN\Machine format).

- MachineUid (System.Int32)

  The UID of the machine to which the connection was made.

- Uid (System.Int64)

  The UID of the connection log entry itself.

Examples

EXAMPLE 1

Gets all connection log entries for sessions brokered in the past 30 minutes, ordered first by machine name (ascending), then by session end time (descending).

```
$when = [DateTime]::Now - [TimeSpan]::FromMinutes(30)
Get-BrokerConnectionLog -Filter {
 BrokeringTime -gt $when }
  -SortBy '+MachineName,-EndTime'
```

Parameters

-Uid

Gets a specific connection log entry identified by its UID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets connection log entries for the specified machines (in DOMAIN\Machine format).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrokeringTime

Gets connection log entries with a specified brokering time. For more flexibility when searching on brokering time use the -Filter parameter.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringUserName

Gets connection log entries for the specified users (in DOMAIN\User format).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrokeringUserUPN

Gets connection log entries for the specified users (in `user@upndomain.com` format).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConnectionFailureReason

Gets connection log entries which failed for the specified reason.

Type:	ConnectionFailureReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Disconnected

Gets connection log entries with the specified disconnection status, that is, whether the connection was disconnected, or logged-off.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndTime

Gets connection log entries with the specified end time. For more flexibility when searching on end time use the -Filter parameter.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EstablishmentTime

Gets connection log entries with the specific establishment time. For more flexibility when searching on establishment time use the -Filter parameter.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MachineDNSName

Gets connection log entries for the specified machines (in `machine@dnsdomain.com` format).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineUid

Gets connection log entries for a specific machine identified by its UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or

descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ConnectionLog

An entry from the connection log.

Related Links

Get-BrokerController

March 11, 2024

Gets Controllers running broker services in the site.

Syntax

```
Get-BrokerController
    [[-MachineName] <String>]
    [-ControllerVersion <String>]
    [-DesktopsRegistered <Int32>]
    [-DNSName <String>]
    [-LastActivityTime <DateTime>]
    [-LastLicensingServerEvent <LicensingServerEvent>]
    [-LastLicensingServerEventTime <DateTime>]
    [-LastStartTime <DateTime>]
    [-LicensingGraceState <LicensingGraceState>]
    [-LicensingServerState <LicensingServerState>]
    [-Metadata <String>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-SID <String>]
    [-State <ControllerState>]
    [-UUID <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerController
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets Controllers from the current site that match the specified search criteria.

Controllers are server machines running instances of the broker service. The broker service is responsible for the brokering of user sessions to desktops or applications, and for power management of the underlying machines. An operational site must contain at least one Controller.

If no search criteria are specified, all Controllers in the site are obtained.

——————————BrokerController Object

The BrokerController object represents a single controller running an instance of the Broker service. It contains the following properties:

- ActiveSiteServices (System.String[])

  The Broker site services active on the controller.

- AssociatedHypervisorConnectionUids (System.Int32[])

  The UIDs of the hypervisor connections being managed by the Broker service on the controller.

- ControllerVersion (System.String)

  The version of the Broker service on the controller.

- DesktopsRegistered (System.Int32)

  The number of VDA machines registered with the Broker service on the controller.

- DNSName (System.String)

  The DNS name of the controller.

- LastActivityTime (System.DateTime?)

  The last reported activity time of the Broker service on the controller.

- LastLicensingServerEvent (Citrix.Broker.Admin.SDK.LicensingServerEvent?)

  Last significant licensing server event reported by the Broker service on the controller.

- LastLicensingServerEventDetails (System.String[])

  Additional details associated with the last significant licensing server event.

- LastLicensingServerEventTime (System.DateTime?)

  Time at which the last significant licensing server event was reported.

- LastStartTime (System.DateTime?)

  The last start-up time of the Broker service on the controller.

- LicensingGracePeriodReasons (Citrix.Broker.Admin.SDK.LicensingGracePeriodReason[])

  Current active or expired licensing grace periods in effect on the controller.

- LicensingGracePeriodTimesRemaining (System.TimeSpan[])

  Times remaining in currently active or expired licensing grace periods in effect on the controller. Expired grace periods are indicated by zero remaining time. The number and order of entries in this list matches that in the LicensingGracePeriodReasons list.

- LicensingGraceState (Citrix.Broker.Admin.SDK.LicensingGraceState)

  The licensing grace state currently in effect in the Broker service on the controller.

- LicensingServerState (Citrix.Broker.Admin.SDK.LicensingServerState)

  The licensing server state currently in effect in the Broker service on the controller.

- MachineName (System.String)

  The Windows name of the controller.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  The metadata for the controller.

- OSType (System.String)

  The Operating System type of the controller.

- OSVersion (System.String)

  The Operating System version of the controller.

- SID (System.String)

  The SID of the controller.

- State (Citrix.Broker.Admin.SDK.ControllerState)

  The state of the Broker service on the controller.

- Uid (System.Int32)

  The UID of the controller instance.

- UUID (System.Guid)

  A globally unique identifier of the controller instance.

Examples

EXAMPLE 1

Gets all Controllers in the site that are currently active (powered on and fully operational).

```
1 Get-BrokerController -State Active
```


EXAMPLE 2

Gets all Controllers in the site that started-up in the last 30 minutes.

```
1  Get-BrokerController -Filter 'LastStartTime -gt "-30:00"'
```

Parameters

-Uid

Gets only Controller with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets only Controllers with the specified Windows name. (‘domain\machine’)

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ControllerVersion

Gets only Controllers running the specified version of the broker service.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopsRegistered

Gets only Controllers that have the specified number of desktops currently registered. This parameter is mainly of use with advanced filtering; see about_Broker_Filtering.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets only Controllers with the specified DNS name ('machine.domain')

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastActivityTime

Gets only Controllers last reported as active at the specified time. This parameter is mainly of use with advanced filtering; see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastLicensingServerEvent

Gets only Controllers with the specified last license server event recorded.

Type:	LicensingServerEvent
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastLicensingServerEventTime

Gets only Controllers with its last recorded licensing server event at the specified time. This parameter is mainly of use with advanced filtering; see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LastStartTime

Gets only Controllers that last started-up at the specified time. This parameter is mainly of use with advanced filtering; see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicensingGraceState

Gets only Controllers in the specified licensing grace state.

Valid values are: NotActive, InOutOfBoxGracePeriod, InSupplementalGracePeriod, InEmergencyGracePeriod and GracePeriodExpired.

Type:	LicensingGraceState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicensingServerState

Gets only Controllers in the specified licensing server state. Valid values are: ServerNotSpecified, NotConnected, OK, LicenseNotInstalled, LicenseExpired, Incompatible and Failed.

Type:	LicensingServerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets only Controllers running the specified Operating System type.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets only Controllers running the specified Operating System version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SID

Gets only Controllers with the specified SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-State

Gets only Controllers currently in the specified state.

Valid values are: Failed, Off, On, and Active.

Type:	ControllerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets only the Controller with the specified GUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Controller

Returns Controllers matching all specified selection criteria.

Related Links

- about_Broker_Concepts
- about_Broker_Desktops
- Get-BrokerDesktop

Get-BrokerDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Broker Service.

Syntax

```
Get-BrokerDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets the database connection string from the currently selected Broker Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Broker Service instance running on controller "controller1.mydomain.net".

```
1 Get-BrokerDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Broker Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the BrokerService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Set-BrokerDBConnection
- Get-BrokerServiceStatus
- Test-BrokerDBConnection

Get-BrokerDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Broker Service.

Syntax

```
Get-BrokerDBSchema
   -DatabaseName <String>
   [-ServiceGroupName <String>]
   [-ScriptType <DatabaseScriptType>]
   [-SID <String>]
   [-LocalDatabase]
   [-DatabaseRights <String>]
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Broker Service database schema, add a new Broker service to an existing site, remove a Broker service from a site, or create a database server logon for a Broker service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Broker service instance, otherwise the scripts relate to Broker service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Broker service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user

- Addition of database user to Broker service roles

If ScriptType is Evict, the returned script contains:

- Removal of Broker service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Broker Service and copies it to a file called "C:\BrokerSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Broker service schema.

```
Get-BrokerDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\BrokerSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Broker service. This can be used when configuring a mirror server for use.

```
Get-BrokerDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    BrokerLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Broker service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Broker services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Broker Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Broker Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Broker Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Broker Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	DatabaseScriptType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Specifies the SID of the controller on which the Broker Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Broker services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Set-BrokerDBConnection
- Test-BrokerDBConnection

Get-BrokerDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Broker Service.

Syntax

```
Get-BrokerDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets an SQL script that can be used to update the current Citrix Broker Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Broker Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Broker Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Broker Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Broker Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-BrokerServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
1 $update = Get-BrokerDBVersionChangeScript -DatabaseName MyDb -
     TargetVersion 7.40.0.0
2 $update.Script > update_740.sql
```


Parameters

-DatabaseName

The name of the database containing the Citrix Broker Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-BrokerDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Broker Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Broker services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-BrokerServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerInstalledDBVersion
- Get-BrokerServiceStatus
- Get-BrokerDBSchema

Get-BrokerDelayedHostingPowerAction

March 11, 2024

Gets power actions that are executed after a delay.

Syntax

```
Get-BrokerDelayedHostingPowerAction
    [[-MachineName] <String>]
    [-Action <PowerManagementAction>]
    [-ActionDueTime <DateTime>]
    [-DNSName <String>]
```



```
    [-HostedMachineName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerDelayedHostingPowerAction
    [-Uid] <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Finds all delayed power actions that match the specified search criteria.

————————BrokerDelayedHostingPowerAction Object

The BrokerDelayedHostingPowerAction object represents an instance of a power action that is executed after a delay. It contains the following properties:

- Action (Citrix.Broker.Admin.SDK.PowerManagementAction)

  The power action to apply to the machine. Possible values are ShutDown and Suspend.

- ActionDueTime (System.DateTime)

  The UTC time at which the power action is due to be queued for execution.

- DNSName (System.String)

  The fully qualified DNS name of the machine that the power action applies to.

- HostedMachineName (System.String)

  The hypervisor's name for the machine that the power action applies to.

- HypervisorConnectionUid (System.Int32)

  The unique identifier of the hypervisor connection that is associated with the target machine.

- MachineName (System.String)

  The name of the machine that the power action applies to, in the form domain\machine.

- Uid (System.Int64)

  The unique identifier of the power action.

Examples

EXAMPLE 1

Fetches records for all known delayed power actions that have not yet been queued for execution.

```
Get-BrokerDelayedHostingPowerAction
```

Parameters

-Uid

Gets only the single action record whose ID matches the specified value.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets only the records for actions that are for machines whose name (of the form domain\machine) matches the specified string.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Action

Gets only the records for actions with the specified action type.

Valid values are Shutdown and Suspend.

Type:	PowerManagementAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ActionDueTime

Gets only the records for actions due to be queued for execution at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets only the records for actions that are for machines whose DNS name matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets only the records for actions that are for machines whose Hosting Name (the machine name as understood by the hypervisor) matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionName

Gets only the records for actions for machines hosted through a hypervisor connection whose name matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets only the records for actions for machines hosted through a hypervisor connection whose ID matches the specified value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DelayedHostingPowerAction

Get-BrokerDelayedHostingPowerAction returns all delayed power actions that match the specified selection criteria.

Related Links

- about_Broker_PowerManagement
- about_Broker_Filtering
- New-BrokerDelayedHostingPowerAction
- Remove-BrokerDelayedHostingPowerAction
- Remove-BrokerHostingPowerAction

Get-BrokerDesktop

March 11, 2024

Gets desktops configured for this site.

Syntax


```
Get-BrokerDesktop
    [[-MachineName] <String>]
    [-AgentVersion <String>]
    [-ApplicationInUse <String>]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserUPN <String>]
    [-AutonomouslyBrokered <Boolean>]
    [-CatalogName <String>]
    [-CatalogUid <Int32>]
    [-ClientAddress <String>]
    [-ClientName <String>]
    [-ClientVersion <String>]
    [-ColorDepth <ColorDepth>]
    [-ConnectedViaHostName <String>]
    [-ConnectedViaIP <String>]
    [-ControllerDNSName <String>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DesktopCondition <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-DesktopKind <DesktopKind>]
    [-DeviceId <String>]
    [-DNSName <String>]
    [-FunctionalLevel <FunctionalLevel>]
    [-HardwareId <String>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-IconUid <Int32>]
    [-ImageOutOfDate <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAssigned <Boolean>]
    [-IsPhysical <Boolean>]
    [-LastConnectionFailure <ConnectionFailureReason>]
    [-LastConnectionTime <DateTime>]
    [-LastConnectionUser <String>]
    [-LastDeregistrationReason <DeregistrationReason>]
    [-LastDeregistrationTime <DateTime>]
    [-LastErrorReason <String>]
    [-LastErrorTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-LaunchedViaHostName <String>]
    [-LaunchedViaIP <String>]
    [-MachineInternalState <MachineInternalState>]
    [-MachineUid <Int32>]
    [-OSType <String>]
```



```
    [-OSVersion <String>]
    [-PersistUserChanges <PersistUserChanges>]
    [-PowerActionPending <Boolean>]
    [-PowerState <PowerState>]
    [-Protocol <String>]
    [-ProvisioningType <ProvisioningType>]
    [-PublishedApplication <String>]
    [-PublishedName <String>]
    [-PvdStage <PvdStage>]
    [-RegistrationState <RegistrationState>]
    [-SecureIcaActive <Boolean>]
    [-SecureIcaRequired <Boolean>]
    [-SessionHidden <Boolean>]
    [-SessionId <Int32>]
    [-SessionState <SessionState>]
    [-SessionStateChangeTime <DateTime>]
    [-SessionUid <Int64>]
    [-SessionUserName <String>]
    [-SessionUserSID <String>]
    [-SID <String>]
    [-SmartAccessTag <String>]
    [-StartTime <DateTime>]
    [-SummaryState <DesktopSummaryState>]
    [-Tag <String>]
    [-WillShutdownAfterUse <Boolean>]
    [-ApplicationUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerDesktop
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet is now deprecated, please use Get-BrokerMachine.

Retrieves desktops matching the specified criteria. If no parameters are specified this cmdlet enumerates all desktops.

Get-BrokerDesktop returns objects that combine desktop configuration and state information.

For single-session desktops, session information is displayed if present. It is possible that there are more than one sessions present on single-session desktops if 'fast user switching'is enabled, this cmdlet will prefer to return information about brokered sessions (rather than, for example, unbrokered direct RDP sessions). If there is no session running, session related fields return $null.

For multi-session desktops, no session information is ever displayed by this cmdlet, so session related fields always return $null. Get-BrokerSession can be used to get information about sessions on both multi-session and single-session desktops.

To count desktops, rather than retrieve full details of each desktop, use Group-BrokerDesktop instead.

For information about advanced filtering options, see about_Broker_Filtering; for information about desktops, see about_Broker_Desktops.

————————BrokerDesktop Object

The desktop object returned represents a physical or virtual machine configured in the site that is able to run either a Microsoft Windows desktop environment, individual applications, or both.

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

- ApplicationsInUse (System.String[])

  List of applications in use on the desktop (in the form of browser name).

- AssignedClientName (System.String)

  The name of the endpoint client device that the desktop has been assigned to.

- AssignedIPAddress (System.String)

  The IP address of the endpoint client device that the desktop has been assigned to.

- AssociatedUserFullNames (System.String[])

  Full names of the users that have been associated with the desktop (in the form "Firstname Lastname").
  Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

- AssociatedUserNames (System.String[])

  Usernames of the users that have been associated with the desktop (usually in the form "domain\user").
  Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

- AssociatedUserUPNs (System.String[])

  The user principal names of the users that have been associated with the desktop (in the form `user@upndomain.com`).
  Associated users are the current user(s) for shared desktops and the assigned users for private desktops.

- AutonomouslyBrokered (System.Boolean?)

  Session property indicating if the current session is an HDX session established by direct connection without being brokered.
  Session properties are always null for multi-session desktops.

- CatalogName (System.String)

  Name of the catalog the desktop is a member of.

- CatalogUid (System.Int32)

  UID of the catalog the desktop is a member of.

- ClientAddress (System.String)

  Session property indicating the IP address of the client connected to the desktop.
  Session properties are always null for multi-session desktops.

- ClientName (System.String)

  Session property indicating the host name of the client connected to the desktop.
  Session properties are always null for multi-session desktops.

- ClientVersion (System.String)

  Session property indicating the version of the Citrix Receiver running on the connected client.
  Session properties are always null for multi-session desktops.

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

  The color depth setting configured on the desktop, possible values are:
  $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

- ConnectedViaHostName (System.String)

  Session property indicating the host name of the connection gateway, router or client.
  Session properties are always null for multi-session desktops.

- ConnectedViaIP (System.String)

  Session property indicating the IP address of the connection gateway, router or client.
  Session properties are always null for multi-session desktops.

- ControllerDNSName (System.String)

  The DNS host name of the controller that the desktop is registered to.

- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType)

  Denotes whether the desktop delivers desktops only, apps only or both.

- Description (System.String)

  Description of the desktop.

- DesktopConditions (System.String[])

  List of outstanding desktop conditions for the desktop.

- DesktopGroupName (System.String)

  Name of the desktop group the desktop has been assigned to.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group the desktop has been assigned to.

- DesktopKind (Citrix.Broker.Admin.SDK.DesktopKind)

  Denotes whether the desktop is private or shared.

- DeviceId (System.String)

  Session property indicating a unique identifier for the client device that has most recently been associated with the current session.
  Session properties are always null for multi-session desktops.

- DNSName (System.String)

  The DNS host name of the desktop.

- FunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel?)

  The functional level of the desktop, if known.

- HardwareId (System.String)

  Session property indicating a unique identifier for the client hardware that has been most recently associated with the current session.
  Session properties are always null for multi-sesison desktops.

- HostedMachineId (System.String)

  Unique ID within the hosting unit of the target managed desktop.

- HostedMachineName (System.String)

  The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

- HostingServerName (System.String)

  DNS name of the hypervisor that is hosting the desktop if managed.

- HypervisorConnectionName (System.String)

  The name of the hypervisor connection that the desktop's hosting server is accessed through, if managed.

- HypervisorConnectionUid (System.Int32?)

  The UID of the hypervisor connection that the desktop's hosting server is accessed through, if managed.

- IconUid (System.Int32?)

  The UID of the desktop's icon that is displayed in StoreFront.

- ImageOutOfDate (System.Boolean?)

  Denotes whether the VM image for a hosted desktop is out of date.

- InMaintenanceMode (System.Boolean)

  Denotes whether the desktop is in maintenance mode.

- IPAddress (System.String)

  The IP address of the desktop.

- IsAssigned (System.Boolean)

  Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explicitly or by assigning on first use of the desktop.

- IsPhysical (System.Boolean)

  This value is true if the desktop is physical(ie not power managed by the Citrix Broker Service), and false otherwise.

- LastConnectionFailure (Citrix.Broker.Admin.SDK.ConnectionFailureReason)

  The reason for the last failed connection between a client and the desktop.

- LastConnectionTime (System.DateTime?)

  Time of the last detected connection attempt that either failed or succeeded.

- LastConnectionUser (System.String)

  The SAM name (in the form DOMAIN\user) of the user that last attempted a connection with the desktop. If the SAM name is not available, the SID is used.

- LastDeregistrationReason (Citrix.Broker.Admin.SDK.DeregistrationReason?)

  The reason for the last deregistration of the desktop with the broker. Possible values are: AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistra-

tionRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- LastDeregistrationTime (System.DateTime?)

  Time of the last deregistration of the desktop from the controller.

- LastErrorReason (System.String)

  The reason for the last error detected in the desktop.

- LastErrorTime (System.DateTime?)

  The time of the last detected error.

- LastHostingUpdateTime (System.DateTime?)

  Time of last update to any hosting data (such as power state) for this desktop reported by the hypervisor connection.

- LaunchedViaHostName (System.String)

  Session property that denotes the host name of the StoreFront server used to launch the current brokered session.
  Session properties are always null for multi-session desktops.

- LaunchedViaIP (System.String)

  Session property that denotes the IP address of the StoreFront server used to launch the current brokered session.
  Session properties are always null for multi-session desktops.

- MachineInternalState (Citrix.Broker.Admin.SDK.MachineInternalState)

  The internal state of the machine associated with the desktop; reported while the desktop is registered to a controller, plus some private Citrix Broker Service states while the machine is not registered.

- MachineName (System.String)

  DNS host name of the machine associated with the desktop.

- MachineUid (System.Int32)

  Uid of the associated machine.

- OSType (System.String)

  A string that can be used to identify the operating system that is running on the desktop.

- OSVersion (System.String)

  A string that can be used to identify the version of the operating system running on the desktop, if known

- PersistUserChanges (Citrix.Broker.Admin.SDK.PersistUserChanges)

  Describes whether/how the user changes are persisted. Possible values are:

  - OnLocal - Persist the user changes on the local disk of the desktop.
  - Discard - Discard user changes.

- PowerActionPending (System.Boolean)

  Property indicating whether there are any pending power actions for the desktop.

- PowerState (Citrix.Broker.Admin.SDK.PowerState)

  The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, resuming.

- Protocol (System.String)

  Session property that denotes the protocol that the current session is using, can be either HDX, RDP or Console. Console sessions on XenDesktop 5 VDAs appear with a blank protocol.
  Session properties are always null for multi-session desktops.

- ProvisioningType (Citrix.Broker.Admin.SDK.ProvisioningType)

  Describes how the machine associated with the desktop was provisioned, possible values are: Manual: No automated provisioning.PVS: Machine provisioned by PVS (may be physical, blade, VM,…)MCS: Machine provisioned by MCS (machine must be VM)

- PublishedApplications (System.String[])

  List of applications published by the desktop (displayed as browser names).

- PublishedName (System.String)

  The name of the desktop that is displayed in StoreFront, if the desktop is published.

- PvdStage (Citrix.Broker.Admin.SDK.PvdStage)

  This property is no longer supported.

- RegistrationState (Citrix.Broker.Admin.SDK.RegistrationState)

  Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

- SecureIcaActive (System.Boolean?)

  Session property that indicates whether SecureICA is active on the current session.
  Session properties are always null for multi-session desktops.

- SecureIcaRequired (System.Boolean?)

  Flag indicating whether SecureICA is required or not when starting a session on the desktop.

- SessionHidden (System.Boolean?)

  Session property that indicates if a session is hidden.
  Session properties are always null for multi-session desktops.

- SessionId (System.Int32?)

  Deprecated. A unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine and only unique at any one particular time.

- SessionState (Citrix.Broker.Admin.SDK.SessionState?)

  Session property indicating the state of the current session.
  Session properties are always null for multi-session desktops, possible values are: Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession and Unknown.

- SessionStateChangeTime (System.DateTime?)

  Session property indicating the time of the last state change of the current session.
  Session properties are always null for multi-session desktops.

- SessionUid (System.Int64?)

  Session property indicating the UID of the current session.
  Session properties are always null for multi-session desktops.

- SessionUserName (System.String)

  Session property indicates the name of the current sessions' user (in the form DOMAIN\user).
  Session properties are always null for multi-session desktops.

- SessionUserSID (System.String)

  Session property indicates the SID of the current sessions' user.
  Session properties are always null for multi-session desktops.

- SID (System.String)

  The SID of the desktop.

- SmartAccessTags (System.String[])

  Session property that indicates the Smart Access tags for the current session.
  Session properties are always null on multi-session desktops.

- StartTime (System.DateTime?)

  Session property that indicates the start time of the current session.
  Session properties are always null on multi-session desktops.

- SummaryState (Citrix.Broker.Admin.SDK.DesktopSummaryState)

  Indicates the overall state of the desktop. The overall state is a result of other more specific states such as session state, registration state and power state. Possible values: Off, Unregistered, Available, Disconnected, InUse, Preparing.

- Tags (System.String[])

  A list of tags for the desktop.

- Uid (System.Int32)

  UID of the desktop object.

- WillShutdownAfterUse (System.Boolean)

  Flag indicating whether this desktop is tainted and will be shut down after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.
  Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Examples

EXAMPLE 1

Both commands retrieve desktops that are unregistered. The second command also includes desktops with a registration state of AgentError.

```
Get-BrokerDesktop -RegistrationState Unregistered
Get-BrokerDesktop -Filter {
  RegistrationState -ne 'Registered'  }
```

EXAMPLE 2

Gets desktops without sessions, listing the DNS name and current state.

```
Get-BrokerDesktop -SessionUid $null | ft -a DNSName,SummaryState
```

EXAMPLE 3

Finds all Windows XP desktops with an out-of-date image.

```
Get-BrokerDesktop -Filter {
  OSType -like "Windows XP*" -and ImageOutOfDate  }
```


EXAMPLE 4

Gets desktops running a published PowerPoint application. It matches any application browser name containing the word 'powerpoint'. String comparisons are case-insensitive.

```
Get-BrokerDesktop -ApplicationInUse '*powerpoint*'
```

EXAMPLE 5

Finds all desktops with an outstanding desktop condition, listing the affected desktop and user.

```
Get-BrokerDesktop -DesktopCondition * DNSName,SessionUserName,
    DesktopConditions
```

EXAMPLE 6

Finds users who have been logged on for more than a day, and outputs the machine name, start time, and duration the session has been logged on.

```
$d = (Get-Date).AddDays(-1)
Get-BrokerDesktop -Filter {
  StartTime -le $d  }
  | ft MachineName,SessionUserName,StartTime,@{
 Label='Duration'; Expression={
 (Get-Date) - $_.StartTime }
  }
```

Parameters

-Uid

Gets desktops with a specific UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets desktops with a specific machine name (in the form 'domain\machine').

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AgentVersion

Gets desktops with a specific Citrix Virtual Delivery Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ApplicationInUse

Gets desktops running a specified published application (identified by browser name).

String comparisons are case-insensitive.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedClientName

Gets desktops assigned to a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedIPAddress

Gets desktops assigned to a specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserFullName

Gets desktops with an associated user identified by their full name (usually in the form ‘first-name last-name’).

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets desktops with an associated user identified by their user name (in the form 'domain\user').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserUPN

Gets desktops with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AutonomouslyBrokered

Gets desktops according to whether their current session is autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets desktops from the catalog with the specific name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogUid

Gets desktops from a catalog with a specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ClientAddress

Gets desktops with a specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientName

Gets desktops with a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientVersion

Gets desktops with a specific client version.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ColorDepth

Gets desktops configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectedViaHostName

Gets desktops with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConnectedViaIP

Gets desktops with a specific IP address of the incoming connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ControllerDNSName

Gets desktops with a specific DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DeliveryType

Gets desktops of a particular delivery type.

Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets desktops with a specific description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopCondition

Gets desktop with an outstanding desktop condition condition.

Valid values are:

- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupName

Gets desktops from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets desktops from a desktop group with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Deprecated: Use AllocationType parameter.

Gets desktops of a particular kind.

Valid values are Private, Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeviceId

Gets desktops with a specific client device ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DNSName

Gets desktops with a specific DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FunctionalLevel

Gets desktops with a specific FunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HardwareId

Gets desktops with a specific client hardware ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineId

Gets desktops with a specific machine ID known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets desktops with a specific machine name known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingServerName

Gets desktops with a specific name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionName

Gets desktops with a specific name of the hosting hypervisor connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets desktops with a specific UID of the hosting hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets desktops with a specific configured icon. Note that desktops with a null IconUid use the icon of the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets desktops by whether their disk image is out of date (for machines provisioned using MCS only).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets desktops with a specific InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets desktops with a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsAssigned

Gets desktops according to whether they are assigned or not. Desktops may be assigned to one or more users or groups, a client IP address or a client endpoint name.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPhysical

Specifies if machines in the catalog can be power managed by the Citrix Broker Service. Where the power state of the machine cannot be controlled, specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the deprecated CatalogKind parameter only with a Pvs catalog kind.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionFailure

Gets desktops with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the desktop yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

Type:	ConnectionFailureReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionTime

Gets desktops that last connected at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionUser

Gets desktops where a specific user name last attempted a connection (in the form 'domain\user').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastDeregistrationReason

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationTime

Gets desktops by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorReason

Gets desktops with the specified last error reason.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastErrorTime

Gets desktops with the specified last error time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets desktops with a specific time that the hosting information was last updated.

Type:	DateTime
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaHostName

Gets desktops with a specific host name of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LaunchedViaIP

Gets desktops with a specific IP address of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineInternalState

Gets desktops with the specified internal machine state.

Type:	MachineInternalState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets desktops with a specific machine UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets desktops by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets desktops by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PersistUserChanges

Gets desktops by the location where the user changes are persisted.

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerActionPending

Gets desktops with a specific power action pending state.

Valid values are $true or $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets desktops with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Protocol

Gets desktops with connections using a specific protocol, for example HDX, RDP, or Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningType

Gets desktops that are in a catalog with a particular provisioning type. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,…).

- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedApplication

Gets desktops with a specific application published to them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PublishedName

Gets desktops with a specific published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PvdStage

This property is no longer supported.

Type:	PvdStage
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RegistrationState

Gets desktops with a specific registration state.

Valid values are Unregistered, Initializing, Registered and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaActive

Gets desktops depending on whether the current session uses SecureICA or not.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-SecureIcaRequired

Gets desktops configured with a particular SecureIcaRequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionHidden

Gets desktops by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionId

Deprecated.

Gets desktops by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Gets desktops with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always null for multi-session desktops.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Gets desktops whose sessions last changed state at a specific time.

Session properties are always null for multi-session desktops.

Type:	DateTime
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets single-session desktops with a specific session UID ($null for no session).

Session properties are always null for multi-session desktops.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserName

Gets desktops with a specific user name for the current session (in the form 'domain\user').

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionUserSID

Gets desktops with a specific SID of the current session user.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SID

Gets desktops with a specific machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SmartAccessTag

Gets session desktops where the session has the specific SmartAccess tag.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-StartTime

Gets desktops with a specific session start time.

Session properties are always null for multi-session desktops.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SummaryState

Gets desktops with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, InUse and Preparing.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets desktops with a specific tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-WillShutdownAfterUse

Gets desktops depending on whether they shut down after use or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets desktops with a specific published application (identified by its UID).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named

Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Desktop

Get-BrokerDesktop returns an object for each matching desktop.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Filtering
- about_Broker_Desktops
- Group-BrokerMachine
- Get-BrokerMachine

Get-BrokerDesktopGroup

March 11, 2024

Gets broker desktop groups configured for this site.

Syntax

```
Get-BrokerDesktopGroup
   [[-Name] <String>]
   [-AdminFolderName <String>]
   [-AdminFolderUid <Int32>]
   [-AllowReconnectInMaintenanceMode <Boolean>]
   [-AppDisk <Guid>]
   [-AppDnaAnalysisState <AppDnaAnalysisState>]
   [-AppDnaCompatibility <AppDnaCompatibility>]
   [-AppProtectionKeyLoggingRequired <Boolean>]
   [-AppProtectionScreenCaptureRequired <Boolean>]
   [-AutomaticPowerOnForAssigned <Boolean>]
   [-AutomaticPowerOnForAssignedDuringPeak <Boolean>]
   [-AutomaticRestartForUntaggedMachines <Boolean>]
   [-AutoscaleLogOffReminderEnabled <Boolean>]
   [-AutoscaleLogOffReminderIntervalSecondsOffPeak <Int32>]
   [-AutoscaleLogOffReminderIntervalSecondsPeak <Int32>]
   [-AutoscaleLogOffReminderMessage <String>]
   [-AutoscaleLogOffReminderTitle <String>]
   [-AutoscaleLogOffWarningMessage <String>]
   [-AutoscaleLogOffWarningTitle <String>]
   [-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak <Int32>]
   [-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak <Int32>]
   [-AutoscalingEnabled <Boolean>]
   [-ColorDepth <ColorDepth>]
```



```
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DesktopGroupName <String>]
    [-DesktopKind <DesktopKind>]
    [-DisconnectOffPeakIdleSessionAfterSeconds <Int32>]
    [-DisconnectPeakIdleSessionAfterSeconds <Int32>]
    [-Enabled <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IsRemotePC <Boolean>]
    [-IsUsedByGpo <Boolean>]
    [-LicenseModel <LicenseModel>]
    [-LogoffOffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-LogoffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-MachineCost <Decimal>]
    [-MachineLogOnType <MachineLogOnType>]
    [-Metadata <String>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-OffPeakBufferSizePercent <Int32>]
    [-OffPeakDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakDisconnectTimeout <Int32>]
    [-OffPeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakExtendedDisconnectTimeout <Int32>]
    [-OffPeakLogOffAction <SessionChangeHostingAction>]
    [-OffPeakLogOffTimeout <Int32>]
    [-PeakAutoscaleAssignedPowerOnIdleAction <String>]
    [-PeakAutoscaleAssignedPowerOnIdleTimeout <Int32>]
    [-PeakBufferSizePercent <Int32>]
    [-PeakDisconnectAction <SessionChangeHostingAction>]
    [-PeakDisconnectTimeout <Int32>]
    [-PeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-PeakExtendedDisconnectTimeout <Int32>]
    [-PeakLogOffAction <SessionChangeHostingAction>]
    [-PeakLogOffTimeout <Int32>]
    [-PolicySetGuid <Guid>]
    [-PowerOffDelay <Int32>]
    [-ProductCode <String>]
    [-PublishedName <String>]
    [-RequiredSleepCapability <String>]
    [-ResourceLeasingEnabled <Boolean>]
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak <Int32>]
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak <Int32>]
    [-RestrictAutoscaleTagUid <Int32>]
    [-ReuseMachinesWithoutShutdownInOutage <Boolean>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-SecureIcaRequired <Boolean>]
    [-SessionSupport <SessionSupport>]
    [-SettlementPeriodBeforeAutoShutdown <TimeSpan>]
    [-SettlementPeriodBeforeUse <TimeSpan>]
    [-ShutdownDesktopsAfterUse <Boolean>]
    [-Tag <String>]
    [-TenantId <Guid>]
    [-TimeZone <String>]
```



```
    [-TotalApplicationGroups <Int32>]
    [-TotalApplications <Int32>]
    [-TurnOnAddedMachine <Boolean>]
    [-UseVerticalScaling <Boolean>]
    [-UUID <Guid>]
    [-ZonePreference <ZonePreference>]
    [-ApplicationGroupUid <Int32>]
    [-ApplicationUid <Int32>]
    [-TagUid <Int32>]
    [-PowerTimeSchemeUid <Int32>]
    [-MachineConfigurationUid <Int32>]
    [-RemotePCCatalogUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerDesktopGroup
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve desktop groups matching the specified criteria. If no parameters are specified this cmdlet enumerates all desktop groups.

Desktop groups represent groups of desktops that are managed together for brokering purposes.

————————BrokerDesktopGroup Object

A desktop group object represents a collection of machines that are fully configured in a site that is able to run either a Microsoft Windows desktop environment, individual applications, or both.

- AdminFolderName (System.String)

  The name of the admin folder the desktop group is in (including trailing backslash), or the empty string if the desktop group is at the root level

- AdminFolderUid (System.Int32)

  The Uid of the admin folder the desktop group is in (if any)

- AllowReconnectInMaintenanceMode (System.Boolean)

  Specifies whether reconnecting sessions are allowed for machines in Maintenance mode.

- AppDisks (System.Guid[])

  The Application Disks used by machines in the desktop group.

- AppDnaAnalysisState (Citrix.Broker.Admin.SDK.AppDnaAnalysisState?)

  Specifies the current state of AppDNA Analysis on the desktop group

- AppDnaCompatibility (Citrix.Broker.Admin.SDK.AppDnaCompatibility?)

  The compatibility with the application disks attached to the desktop group

- AppProtectionKeyLoggingRequired (System.Boolean)

  Specifies whether key logging app protection is required.

- AppProtectionScreenCaptureRequired (System.Boolean)

  Specifies whether screen capture app protection is required.

- AutomaticPowerOnForAssigned (System.Boolean)

  Specifies whether assigned desktops in the desktop group are automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

- AutomaticPowerOnForAssignedDuringPeak (System.Boolean)

  Specifies whether assigned desktops in the desktop are automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

- AutomaticRestartForUntaggedMachines (System.Boolean)

  Indicates whether untagged single-session machines belonging to a desktop group configured for restrict Autoscale and shutdown after use should restart after untainting.

- AutoscaleLogOffReminderEnabled (System.Boolean)

  Boolean value indicating whether the warning messages should be sent on an interval to nudge a logoff should be sent on an interval when autoscale is enabled.

- AutoscaleLogOffReminderIntervalSecondsOffPeak (System.Int32)

  Represents the time interval at which messages are sent to the user during off peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off.

- AutoscaleLogOffReminderIntervalSecondsPeak (System.Int32)

  Represents the time interval at which messages are sent to the user during peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off.

- AutoscaleLogOffReminderMessage (System.String)

  Notification message to display to users in active sessions belonging to machines needed by Autoscale for shutdown.

- AutoscaleLogOffReminderTitle (System.String)

  Notification message dialog title displayed when Autoscale issues a logoff reminder request.

- AutoscaleLogOffWarningMessage (System.String)

  Warning message to display to users in active sessions prior to Autoscale issuing a logoff request.

- AutoscaleLogOffWarningTitle (System.String)

  Warning message dialog title displayed prior to Autoscale issuing a logoff request.

- AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak (System.Int32)

  Minimum time that needs to elapse before Autoscale logs off active sessions on a draining machine in the delivery group during off-peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

- AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak (System.Int32)

  Minimum time that needs to elapse before Autoscale logs off active sessions on a draining machine in the delivery group during peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

- AutoscalingEnabled (System.Boolean)

  Specifies whether machines in this desktop group can be Autoscaled.

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth)

  Default color depth of sessions started with machines in the desktop group. Possible values are: FourBit, EightBit, SixteenBit, TwentyFourBit.

- ConfigurationSlotUids (System.Int32[])

  Uids of any configuration slots which hold machine configurations associated with the desktop group. The order of slot UIDs in this list correspond with the order of items in the associated MachineConfigurationNames and MachineConfigurationUids list properties, and so the same slot UID can appear more than once.

- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType)

  The type of resources being published. Possible values are:
  DesktopsOnly, AppsOnly, DesktopsAndApps.

- Description (System.String)

  Description of the desktop group.

- DesktopGroupName (System.String)

  Name of the desktop group (must be unique within a Folder)

- DesktopKind (Citrix.Broker.Admin.SDK.DesktopKind)

  The kind of the desktops being published, possible values are:
  Private and Shared.

- DesktopsAvailable (System.Int32)

  The number of machines in the desktop group in state Available; this is the number of machines with no sessions present.

- DesktopsDisconnected (System.Int32)

  The number of disconnected sessions present on machines in the desktop group.

- DesktopsFaulted (System.Int32)

  The number of machines in the desktop group whose FaultState is not None.

- DesktopsInUse (System.Int32)

  The number of machines in the desktop group in state InUse; this is the number of machines with at least one session present.

- DesktopsNeverRegistered (System.Int32)

  The number of machines in the desktop group that have never registered with the current site.

- DesktopsPreparing (System.Int32)

  This property is no longer supported.

- DesktopsUnregistered (System.Int32)

  The number of machines in the desktop group that are currently unregistered.

- DisconnectOffPeakIdleSessionAfterSeconds (System.Int32)

  Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during off-peak time.

- DisconnectPeakIdleSessionAfterSeconds (System.Int32)

  Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during peak time.

- Enabled (System.Boolean)

  Specifies whether the desktop group is enabled or not; disabled desktop groups do not appear to users.

- IconUid (System.Int32)

  The Uid of the icon to be used as a default for desktops in the desktop group. Individual desktop objects can override this default by setting the IconUid parameter on the desktop object.

- InMaintenanceMode (System.Boolean)

  Specifies whether the machines in the desktop group are in maintenance mode or not.

- IsRemotePC (System.Boolean)

  Specifies whether the desktop group is a Remote PC desktop group.

- IsUsedByGpo (System.Boolean?)

  Indicate if the desktop group is used by group policy.

- LicenseModel (Citrix.Broker.Admin.SDK.LicenseModel?)

  Specifies the license model for this desktop group. If none is specified, then the site-wide license model is used.

- LogoffOffPeakDisconnectedSessionAfterSeconds (System.Int32)

  Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during off-peak time.

- LogoffPeakDisconnectedSessionAfterSeconds (System.Int32)

  Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during peak time.

- MachineConfigurationNames (System.String[])

  The MachineConfiguration names associated with the desktop group.

- MachineConfigurationUids (System.Int32[])

  The MachineConfiguration uids associated with the desktop group.

- MachineCost (System.Decimal)

  The per-hour instance cost of machines in this desktop group.

- MachineLogOnType (Citrix.Broker.Admin.SDK.MachineLogOnType?)

  Specifies the login type (activedirectory/localmappedaccount) for machines in this desktop group.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata associated with the desktop group.

- MinimumFunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel)

  The minimum FunctionalLevel required for the machines in the desktop group to be able to register with the Citrix Broker Service.

- Name (System.String)

  Site-wide unique full path name of the desktop group including all containing folders, for example, Folder1\Folder2\MyDesktopGroup.

- OffPeakBufferSizePercent (System.Int32)

  The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available outside peak hours.

- OffPeakDisconnectAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action that is performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend or Shutdown.

- OffPeakDisconnectTimeout (System.Int32)

  The number of minutes before the configured action is performed after a user session disconnects outside peak hours.

- OffPeakExtendedDisconnectAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

- OffPeakExtendedDisconnectTimeout (System.Int32)

  The number of minutes before the second configured action is performed after a user session disconnects outside peak hours.

- OffPeakLogOffAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action performed after a configurable period of a user session ending outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

- OffPeakLogOffTimeout (System.Int32)

  The number of minutes before the configured action is performed after a user session ends outside peak hours.

- PeakAutoscaleAssignedPowerOnIdleAction (System.String)

  The action to be performed on an assigned machine started by Autoscale if that machine then remains unused for a defined period of time.

- PeakAutoscaleAssignedPowerOnIdleTimeout (System.Int32)

  The number of minutes before the configured action is performed on an assigned machine previously started by autoscale that subsequently remains unused.

- PeakBufferSizePercent (System.Int32)

The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available in peak hours.

- PeakDisconnectAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action performed after a configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

- PeakDisconnectTimeout (System.Int32)

  The number of minutes before the configured action is performed after a user session disconnects in peak hours.

- PeakExtendedDisconnectAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action performed after a second configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

- PeakExtendedDisconnectTimeout (System.Int32)

  The number of minutes before the second configured action is performed after a user session disconnects in peak hours.

- PeakLogOffAction (Citrix.Broker.Admin.SDK.SessionChangeHostingAction)

  The action performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

- PeakLogOffTimeout (System.Int32)

  The number of minutes before the configured action is performed after a user session ends in peak hours.

- PolicySetGuid (System.Guid?)

  The policy set assigned to this delivery group. If this is null, the policy set of the site is used.

- PowerOffDelay (System.Int32)

  The number of minutes following a power-on operation that Auto Scale will wait before attempting to power off that same machine. Possible values are in the range 0 - 60.

- ProductCode (System.String)

  Specifies the licensing product code for this desktop group. If none is specified, then the site-wide product code is used.

- ProtocolPriority (System.String[])

  A list of protocol names in the order in which they are attempted for use during connection.

- PublishedName (System.String)

  The name of the desktop group as it is to appear to the user in StoreFront.

- RequiredSleepCapability (System.String)

  The sleep capability of this desktop group. Possible values are None, or Suspend.

- ResourceLeasingEnabled (System.Boolean)

  Indicates if this desktop group is enabled for resource leasing.

- RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak (System.Int32)

  This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages 'tagged'machines, as per policy, in off-peak. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down 'tagged'machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for desktop groups configured for Restrict Autoscaling.

- RestrictAutoscaleMinIdleUntaggedPercentDuringPeak (System.Int32)

  This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages 'tagged'machines, as per policy, in peak time. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down 'tagged'machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for desktop groups configured for Restrict Autoscaling.

- RestrictAutoscaleTagUid (System.Int32?)

  If set to a Tag UID, only machines that have this tag will be auto scaled.

- ReuseMachinesWithoutShutdownInOutage (System.Boolean)

  Specifies whether desktops should not be shut down after being used during outage mode.

- Scopes (Citrix.Fma.Sdk.DelegatedAdminInterfaces.ScopeReference[])

  The list of the delegated admin scopes to which the desktop group belongs.

- SecureIcaRequired (System.Boolean)

  Flag that specifies if the SecureICA encryption of the HDX protocol is required for sessions of desktops in the desktop group.

- Sessions (System.Int32)

  The total number of user sessions currently running on all of the machines in the desktop group.

- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport)

Specifies the session support (single/multi) of the machines in the desktop group. Machines with the incorrect session support for the desktop group will be unable to register with the Citrix Broker Service.

- SettlementPeriodBeforeAutoShutdown (System.TimeSpan)

  Time after a session ends during which automatic shutdown requests (for example, shutdown after use, idle pool management) are deferred. Any outstanding shutdown request takes effect after the settlement period expires. This is typically used to configure time to allow logoff scripts to complete.

- SettlementPeriodBeforeUse (System.TimeSpan)

  Idle period before a machine can be selected to host a new session after registration or the end of a previous session. This is typically used to allow a machine to become idle following processing associated with start-up or logoff actions. A machine may still be selected during the idle period if no other machine is available for immediate use.

- ShutdownDesktopsAfterUse (System.Boolean)

  Specifies if the desktops will shut down after they have been used and there are no sessions running on the machine. The machines will not shut down if they are placed into maintenance mode, even if this flag is set to $true. The machines, however, will shutdown after the machine is taken out of maintenance mode if the flag is still set.

- Tags (System.String[])

  Tags associated with the desktop group.

- TenantId (System.Guid?)

  Identity of tenant associated with desktop group. Not applicable (always blank) in non-multitenant sites.

- TimeZone (System.String)

  The timezone that desktops in the desktop group are in (for power policy purposes).

- TotalApplicationGroups (System.Int32)

  Total number of application groups associated with the desktop group.

- TotalApplications (System.Int32)

  Total number of applications associated with the desktop group.

- TotalDesktops (System.Int32)

  Total number of machines in the desktop group.

- TurnOnAddedMachine (System.Boolean)

Specifies whether the broker should attempt to turn on power-managed machines when they are added to the desktop group.

- Uid (System.Int32)

  Uid of the desktop group.

- UseVerticalScaling (System.Boolean?)

  Indicates whether to use vertical scaling when finding an RDS machine in the desktop group for a session launch.

- UUID (System.Guid)

  UUID of the desktop group.

- ZonePreferences (Citrix.Broker.Admin.SDK.ZonePreference[])

  Ordered list of zone preferences taken into account when launching resources from this desktop group. Possible values for each preference are UserLocation, UserHome, UserHomeOnly and ApplicationHome.

Examples

EXAMPLE 1

Finds all desktop groups with published names starting with "EMEA".

```
1  Get-BrokerDesktopGroup -PublishedName EMEA*
```

EXAMPLE 2

Finds all desktop groups in maintenance mode.

```
1  Get-BrokerDesktopGroup -InMaintenanceMode $true
```

Parameters

-Uid

Gets desktop groups with the specified value of Uid.

Type:	Int32
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets desktop groups whose full path name matches the supplied pattern.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderName

Gets desktop groups that are in admin folders matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminFolderUid

Gets desktop groups that are in the specified admin folder.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowReconnectInMaintenanceMode

Gets desktop groups which allows session reconnection while in Maintenance mode.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDisk

Gets only desktop groups using the specified application disk.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDnaAnalysisState

Gets only desktop groups with specified value of AppDnaAnalysisState

Type:	AppDnaAnalysisState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDnaCompatibility

Gets only desktop groups with specified value of AppDnaCompatibility

Type:	AppDnaCompatibility
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionKeyLoggingRequired

Specifies whether key logging app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionScreenCaptureRequired

Specifies whether screen capture app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticPowerOnForAssigned

Gets only desktop groups with the specified value of AutomaticPowerOnForAssigned.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticPowerOnForAssignedDuringPeak

Gets only desktop groups with the specified value of AutomaticPowerOnForAssignedDuringPeak.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticRestartForUntaggedMachines

Gets desktop groups with the specified value of AutomaticRestartForUntaggedMachines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderEnabled

Gets the value indicating whether the warning messages should be sent on an interval to nudge a logoff should be sent on an interval when autoscale is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsOffPeak

Gets the interval in seconds which represents the time interval at which messages are sent to the user during off peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsPeak

Gets the interval in seconds which represents the time interval at which messages are sent to the user during peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderMessage

Gets the notification message to display to users in active sessions belonging to machines needed by Autoscale for shutdown.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AutoscaleLogOffReminderTitle

Gets the notification message dialog title displayed when Autoscale issues a logoff reminder request.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-AutoscaleLogOffWarningMessage

Gets the warning message to display to users in active sessions prior to Autoscale issuing a logoff request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AutoscaleLogOffWarningTitle

Gets the warning message dialog title displayed prior to Autoscale issuing a logoff request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak

Gets the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group during off-peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak

Gets the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group during peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscalingEnabled

Specifies whether machines in this desktop group can be Autoscaled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Gets only desktop groups with the specified color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeliveryType

Gets desktop groups according to their delivery type.

Valid values are DesktopsOnly, AppsOnly and DesktopsAndApps.

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets desktop groups whose description matches the supplied pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-DesktopGroupName

Gets Desktop groups that match the specified simple name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopKind

Gets desktops of a particular kind.

Valid values are Private and Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisconnectOffPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during off-peak time.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisconnectPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets desktop groups with the specified value of Enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets desktop groups with the specified value of InMaintenanceMode.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsRemotePC

Gets desktop groups with the specified IsRemotePC value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsUsedByGpo

Gets desktop groups that are used by group policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseModel

Gets desktop groups with the specified license model.

Type:	LicenseModel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffOffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during off-peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineCost

The per-hour instance cost of machines in this desktop group.

Type:	Decimal
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineLogOnType

Gets desktop groups that have the specified login type. Values can be:

- ActiveDirectory - Perform Active Directory login on the machine.
- LocalMappedAccount - Perform local mapped account login on the machine.

Type:	MachineLogOnType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MinimumFunctionalLevel

Gets desktop groups with a specific MinimumFunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakBufferSizePercent

Gets desktop groups with the specified value of OffPeakBufferSizePercent.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakDisconnectAction

Gets desktop groups with the specified value of OffPeakDisconnectAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakDisconnectTimeout

Gets desktop groups with the specified value of OffPeakDisconnectTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakExtendedDisconnectAction

Gets desktop groups with the specified value of OffPeakExtendedDisconnectAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakExtendedDisconnectTimeout

Gets desktop groups with the specified value of OffPeakExtendedDisconnectTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakLogOffAction

Gets desktop groups with the specified value of OffPeakLogOffAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakLogOffTimeout

Gets desktop groups with the specified value of OffPeakLogOffTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleAction

Gets desktop groups with the specified value of PeakAutoscaleAssignedPowerOnIdleAction.

Type:	String
Accepted values:	Nothing, Shutdown, Suspend
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleTimeout

Gets desktop groups with the specified value of PeakAutoscaleAssignedPowerOnIdleTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakBufferSizePercent

Gets desktop groups with the specified value of PeakBufferSizePercent.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakDisconnectAction

Gets desktop groups with the specified value of PeakDisconnectAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakDisconnectTimeout

Gets desktop groups with the specified value of PeakDisconnectTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakExtendedDisconnectAction

Gets desktop groups with the specified value of PeakExtendedDisconnectAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakExtendedDisconnectTimeout

Gets desktop groups with the specified value of PeakExtendedDisconnectTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakLogOffAction

Gets desktop groups with the specified value of PeakLogOffAction.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakLogOffTimeout

Gets desktop groups with the specified value of PeakLogOffTimeout.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicySetGuid

Gets the policy set assigned to this delivery group.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerOffDelay

The number of minutes following a power-on operation that Auto Scale will wait before attempting to power off that same machine. Possible values are in the range 0 - 60.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductCode

Gets desktop groups with the specified licensing product code.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PublishedName

Gets desktop groups whose published name matches the supplied pattern.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RequiredSleepCapability

Gets desktop groups with the specified value of RequiredSleepCapability.

Type:	String
Accepted values:	None, Suspend
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceLeasingEnabled

Gets desktop groups with the specified value of ResourceLeasingEnabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak

Gets desktop groups with the specified value of RestrictAutoscaleMinIdleUntaggedPercentDuringOff-Peak.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak

Gets desktop groups with the specified value of RestrictAutoscaleMinIdleUntaggedPercentDuring-Peak.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleTagUid

If set to a Tag UID, only machines that have this tag will be auto scaled.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ReuseMachinesWithoutShutdownInOutage

Gets only desktop groups that won't shut down machines after they been used during outage.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets desktop groups that are associated with the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets desktop groups that are associated with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-SecureIcaRequired

Gets desktop groups with the specified value of SecureIcaRequired.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSupport

Gets desktop groups that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettlementPeriodBeforeAutoShutdown

Gets desktop groups with the specified value of SettlementPeriodBeforeAutoShutdown.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettlementPeriodBeforeUse

Gets desktop groups with the specified value of SettlementPeriodBeforeUse.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShutdownDesktopsAfterUse

Gets desktop groups with the specified value of ShutdownDesktopsAfterUse.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets desktop groups tagged with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantId

Gets desktop groups associated with the specified tenant identity.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TimeZone

Gets desktop groups with the specified value of TimeZone.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TotalApplicationGroups

Gets desktop groups that are acting as delivery groups for the specified number of application groups.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TotalApplications

Gets desktop groups that are acting as delivery groups for the specified number of applications.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TurnOnAddedMachine

Gets desktop groups with the specified value of TurnOnAddedMachine value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseVerticalScaling

Gets desktop groups with the specified value of UseVerticalScaling.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets desktop groups with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZonePreference

Gets desktop groups with a zone preference list containing the specified zone preference.

Type:	ZonePreference
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationGroupUid

Gets only desktop groups with which the specified application group has been associated.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets desktop groups that publish the specified application (identified by Uid)

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TagUid

Gets desktop groups to which the specified tag (identified by its Uid) has been added to help identify it - see Add-BrokerTag for more information.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerTimeSchemeUid

Gets desktop groups associated with the specified power time scheme (identified by its Uid).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineConfigurationUid

Gets desktop groups with the specified value of MachineConfiguration.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemotePCCatalogUid

Gets Remote PC desktop groups associated with the specified catalog.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DesktopGroup

Get-BrokerDesktopGroup returns an object for each matching desktop group.

Notes

To perform greater-than or less-than comparisons, use -Filter. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Filtering
- about_Broker_Desktops
- about_Broker_Machines
- about_Broker_RemotePC
- New-BrokerDesktopGroup
- Set-BrokerDesktopGroup
- Rename-BrokerDesktopGroup
- Move-BrokerDesktopGroup
- Remove-BrokerDesktopGroup
- Add-BrokerUser
- Add-BrokerTag

Get-BrokerDesktopGroupAnalysisReport

March 11, 2024

Gets the detailed AppDNA compatibility report for AppDisk(s) associated with a particular Desktop Group.

Syntax

```
Get-BrokerDesktopGroupAnalysisReport
    [-InputObject] <DesktopGroup[]>
    [[-AppDiskUid] <Guid>]
    [-RetrieveReportContentsAsMHT]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the detailed AppDNA compatibility report for an AppDisk associated with a particular Desktop Group.

Examples

EXAMPLE 1

Retrieves the report for the desktop group with Uid 36

```
Get-BrokerDesktopGroupAnalysisReport -InputObject 36
```

EXAMPLE 2

Retrieves the report for the desktop group with Uid 36 filtered down to AppDisk “MyAppDisk”

```
$appDisk = Get-AppLibAppDisk -AppDiskName "MyAppDisk"
Get-BrokerDesktopGroupAnalysisReport -InputObject 36 -AppDiskUid
    $appDisk.AppDiskUid
```

Parameters

-InputObject

The desktop group(s) on which to retrieve the detailed reports

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-RetrieveReportContentsAsMHT

Retrieve the report as a MHT file

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDiskUid

AppDisk unique identifier. If set, the report will be filtered down to only include this AppDisk.

Type:	Guid
Position:	3
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

You can pipe the desktop groups to Get-BrokerDesktopGroupAnalysisReport.

Outputs

Citrix.Broker.Admin.SDK.AnalysisReport

The analysis reports for the desktop groups

Related Links

- Get-BrokerDesktopGroup
- Set-BrokerDesktopGroup

Get-BrokerDesktopGroupAppDisk

March 11, 2024

Gets the AppDisks that are being used by desktop group

Syntax

```
Get-BrokerDesktopGroupAppDisk
   [[-DesktopGroupName] <String>]
   [-AppDiskUid <Guid>]
   [-AppDnaCompatibility <AppDnaCompatibility>]
   [-DesktopGroupUid <Int32>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
```



```
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerDesktopGroupAppDisk cmdlet returns all the AppDisks that are currently being used by the desktop groups and if they are compatible with their desktop group

————————————BrokerDesktopGroupAppDisk Object

The BrokerDesktopGroupAppDisk object represents a single instance of a desktop group and AppDisk and if they are compatible

- AppDiskUid (System.Guid)

  The Uid of the AppDisk

- AppDnaCompatibility (Citrix.Broker.Admin.SDK.AppDnaCompatibility?)

  Specifies if the AppDisk is compatible with the desktop group

- DesktopGroupName (System.String)

  The name of the desktop group

- DesktopGroupUid (System.Int32)

  The Uid of the desktop group

Examples

EXAMPLE 1

Gets all the AppDisks that are assocated to the desktop group MyGroup

```
Get-BrokerDesktopGroupAppDisk -DesktopGroupName MyGroup
```

Parameters

-DesktopGroupName

Gets only the entries that match the specified Desktop Group Name

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AppDiskUid

Gets only the entries that match the specified AppDisk Uid

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDnaCompatibility

Gets only the entries that match the specified AppDnaCompatibility

Type:	AppDnaCompatibility
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets only the entries that match the specified Desktop Group Uid

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DesktopGroupAppDisk

Returns an object for each enumerated Desktop Group and AppDisk combination.

Related Links

- Get-BrokerDesktopGroup

Get-BrokerDesktopGroupWebhook

March 11, 2024

Gets the webhook configured for desktop group

Syntax

```
Get-BrokerDesktopGroupWebhook
    [-Address <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-OnEvent <WebhookTrigger>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
```



```
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerDesktopGroupWebhook
   -Uid <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The cmdlet returns the webhook that are currently being configured for desktop groups. Each desktop group can have up to one webhook.

——————————BrokerDesktopGroupWebhook Object

The BrokerDesktopGroupWebhook object represents a webhook of a desktop group that is invoked upon certain events, currently the only supported event is MachineRegistration.

- Address (System.String)

  URL of the webhook

- DesktopGroupName (System.String)

  The name of the desktop group

- DesktopGroupUid (System.Int32)

  The Uid of the desktop group

- OnEvent (Citrix.Broker.Admin.SDK.WebhookTrigger)

  The event upon that the webhook is triggered

- Uid (System.Int32)

  The Uid of the webhook

Examples

EXAMPLE 1

Gets the webhook that is associated to the desktop group 1

```
Get-BrokerDesktopGroupWebhook -DesktopGroupUid 1
```


Parameters

-Uid

Get only webhooks that match the specified Webhook Uid

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Address

Gets only webhooks whose URL matches that specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupName

Gets only webhooks associated with desktop groups whose names match that specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-DesktopGroupUid

Gets only webhooks associated with the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OnEvent

Gets only webhooks that match the specified OnEvent.

Type:	WebhookTrigger
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named

Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet

Outputs

Citrix.Broker.Admin.SDK.DesktopGroupWebhook

Returns an object for each enumerated Desktop Group and an Webhook combination

Related Links

- Get-BrokerDesktopGroup

Get-BrokerDesktopUsage

March 11, 2024

Get usage history of desktop groups.

Syntax

```
Get-BrokerDesktopUsage
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-InUse <Int32>]
    [-Timestamp <DateTime>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns information, recorded by the broker on an hourly basis, about the number of desktops in use for each desktop group. Analyzing the historical usage records can give some guidance on usage patterns and help choosing idle pool settings.

Without parameters, Get-BrokerDesktopUsage returns the first 250 records. By using parameters, you can be more selective about the records that are returned.

To retrieve more than the default 250 records, use the -MaxRecordCount parameter. To select data for a specific desktop group, use either the -DesktopGroupName or -DesktopGroupUid parameters.

See the examples for this cmdlet and about_Broker_Filtering for details of how to perform advanced filtering.

————————BrokerDesktopUsage Object

Desktop usage object contains information to tell how many desktops in a desktop group are in use at a given time (identified by a timestamp).

- DesktopGroupUid (System.Int32)

  Uid of the desktop group that the usage data corresponds to.

- InUse (System.Int32)

  Specifies how many desktop are in use at the time the timestamp corresponds to.

- Timestamp (System.DateTime)

  Date and time the desktop usage information corresponds to.

Examples

EXAMPLE 1

Returns the last 24 hours of usage information for desktop group TestGroup, formatting it as two columns labeled Time and InUse.

```
Get-BrokerDesktopUsage -DesktopGroupName TestGroup -MaxRecordCount 24 -
    SortBy '-Timestamp' | ft -a @{
 Name='Time';Expression={
 '{
 0:t }
 ' -f $_.Timestamp }
  }
 ,InUse
```

EXAMPLE 2

Returns today's usage information for desktop group TestGroup.

```
$d = Get-Date -Hour 0 -Minute 0 -Second 0
Get-BrokerDesktopUsage -Filter {
  DesktopGroupName -eq 'TestGroup' -and Timestamp -ge $d  }
```

EXAMPLE 3

Retrieves the usage history for desktop group TestGroup and adds a column showing the number of desktops in that group in use, as a percentage.

```
$dg = Get-BrokerDesktopGroup TestGroup
Get-BrokerDesktopUsage -DesktopGroupUid $dg.Uid | Select Timestamp,
    InUse,@{
```



```
Name='Percent';Expression={
'{
0:P0 }
' -f ($_.InUse / $dg.TotalDesktops) }
 }
```

Parameters

-DesktopGroupName

Gets usage records for the named desktop group or for multiple desktop groups if wildcards have been specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets usage records for a specific desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InUse

Gets usage records where the in-use count matches the specified value. This is useful when checking for zero or when used inside a -Filter expression.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Timestamp

Gets usage records that occurred at the given time.

In general, Citrix recommends, using -Filter and relative comparisons. For a demonstration, see the examples.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DesktopUsage

Get-BrokerDesktopUsage returns an object for each matching record.

Notes

Desktop usage information is automatically deleted after 7 days.

Related Links

- about_Broker_Filtering

Get-BrokerEntitlementPolicyRule

March 11, 2024

Gets desktop rules from the site's entitlement policy.

Syntax

```
Get-BrokerEntitlementPolicyRule
    [[-Name] <String>]
    [-BrowserName <String>]
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    [-DesktopGroupUid <Int32>]
    [-Enabled <Boolean>]
    [-ExcludedUser <User>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-IconUid <Int32>]
    [-IncludedUser <User>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-LeasingBehavior <LeasingBehavior>]
    [-MaxPerEntitlementInstances <Int32>]
    [-Metadata <String>]
    [-PublishedName <String>]
    [-RestrictToTag <String>]
    [-SecureIcaRequired <Boolean>]
    [-SessionReconnection <SessionReconnection>]
    [-UUID <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerEntitlementPolicyRule
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Returns desktop rules matching the specified search criteria from the site's entitlement policy. If no search criteria are specified, all desktop rules in the entitlement policy are obtained.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

——————————BrokerEntitlementPolicyRule Object

The BrokerEntitlementPolicyRule object represents a single desktop rule within the site's entitlement policy. It contains the following properties:

- BrowserName (System.String)

  Site-wide unique name identifying this desktop entitlement to other components (for example StoreFront).

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

  The color depth of any desktop session launched by the user from the entitlement. If null, the equivalent setting from the rule's desktop group is used.

- Description (System.String)

  Optional description of the rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

- DesktopGroupUid (System.Int32)

  The unique ID of the desktop group to which the rule applies.

- Enabled (System.Boolean)

  Indicates whether the rule is enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

- ExcludedUserFilterEnabled (System.Boolean)

  Indicates whether the excluded users filter is enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

- ExcludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

- IconUid (System.Int32?)

  The unique ID of the icon used to display the desktop entitlement to the user. If null, the equivalent setting from the rule's desktop group is used.

- IncludedUserFilterEnabled (System.Boolean)

  Indicates whether the included users filter is enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a desktop session by the rule.

- IncludedUsers (Citrix.Broker.Admin.SDK.ChbUser[])

  The included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

- LeasingBehavior (Citrix.Broker.Admin.SDK.LeasingBehavior)

  Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:
  Allowed and Disallowed.

- MaxPerEntitlementInstances (System.Int32)

  Maximum allowed concurrently running instances of the desktop associated with this entitlement in the site . A value of zero allows unlimited usage.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  A collection of arbitrary key/value pairs that can be associated with the rule. The administrator can use these values for any purpose; they are not used by the site itself in any way.

- Name (System.String)

  The administrative name of the rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

- PublishedName (System.String)

  The name of the desktop session entitlement as seen by the user. If null, the equivalent setting from the rule's desktop group is used.

- RestrictToTag (System.String)

  Optional tag that may be used further to restrict which machines may be made accessible to a user by an entitlement policy rule. A machine may be made accessible by an entitlement policy rule only if either the rule has no tag restriction or the rule does have a tag restriction and the machine is tagged with the same tag.

- SecureIcaRequired (System.Boolean?)

  Indicates whether the rule requires the SecureICA protocol for desktop sessions launched using the entitlement. If null, the equivalent setting from the rule's desktop group is used.

- SessionReconnection (Citrix.Broker.Admin.SDK.SessionReconnection)

Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:
Always, DisconnectedOnly, and SameEndpointOnly.

- Uid (System.Int32)

  The unique ID of the rule itself.

- UUID (System.Guid)

  UUID of the rule.

Examples

EXAMPLE 1

Returns all desktop rules from the entitlement policy. This offers a complete description of the current site's entitlement policy with respect to desktops published from shared desktop groups.

```
Get-BrokerEntitlementPolicyRule
```

EXAMPLE 2

Returns all desktop rules in the entitlement policy that give users entitlements to desktop sessions in the Customer Support desktop group.

```
$dg = Get-BrokerDesktopGroup 'Customer Support'
Get-BrokerEntitlementPolicyRule -DesktopGroupUid $dg.Uid
```

Parameters

-Uid

Gets the desktop rule with the specified unique ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only desktop rules with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrowserName

Gets only desktop rules with browser names matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ColorDepth

Gets only desktop rules with the specified color depth.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets only desktop rules with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only desktop rules that apply to the desktop group with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets only desktop rules that are in the specified state, either enabled ($true), or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUser

Gets only desktop rules that have the specified user in their excluded users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Gets only desktop rules that have their excluded user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets only desktop rules using the icon with the specified unique ID.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUser

Gets only desktop rules that have the specified user in their included users filter (whether the filter is enabled or not).

Type:	User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Gets only desktop rules that have their included user filter enabled ($true) or disabled ($false).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeasingBehavior

Gets only application rules with the specified connection leasing behavior. Possible values are:

Allowed and Disallowed.

Type:	LeasingBehavior
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerEntitlementInstances

Maximum allowed concurrently running instances of the desktop associated with this entitlement in the site . A value of zero allows unlimited usage.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-PublishedName

Gets only desktop rules with the specified published name, that is, the desktop session entitlement name that the end user sees.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RestrictToTag

Gets only desktop rules with the specified tag restriction.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SecureIcaRequired

Gets only desktop rules that require the desktop session to use the SecureICA protocol ($true) or not ($false).

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Gets only desktop rules with the specified session reconnection behavior. Possible values are: Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets rules with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

Get-BrokerEntitlementPolicyRule returns all desktop entitlement policy rules that match the specified selection criteria.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_ErrorHandling
- New-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule

Get-BrokerGpoFilter

March 11, 2024

Gets GPO filters for a policy.

Syntax

```
Get-BrokerGpoFilter
   [-FilterData <String>]
   [-FilterType <String>]
   [-IsAllowed <Boolean>]
   [-IsEnabled <Boolean>]
   [-PolicyGuid <Guid>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Get-BrokerGpoFilter
   -FilterGuid <Guid>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieve GPO filters that meet certain criteria.

————————BrokerGpoFilter Object

A GPO filter specifies how the policy should be applied based on the properties of a user session and/or the VDA hosting the session.

- FilterData (System.String)

  Filter data.

- FilterGuid (System.Guid)

  The GUID of the filter.

- FilterType (System.String)

  The type of the filter.

- IsAllowed (System.Boolean)

  Is filter mode set to allowed.

- IsEnabled (System.Boolean)

  Is filter enabled.

- PolicyGuid (System.Guid)

  The GUID of the policy that owns the filter.

Examples

EXAMPLE 1

Gets the filters defined for a policy.

```
Get-BrokerGpoFilter -PolicyGuid ([Guid]"12345678-...")
```


EXAMPLE 2

Gets a specific filter.

```
Get-BrokerGpoFilter -FilterGuid ([Guid]"12345678-...")
```

Parameters

-FilterGuid

Gets specified filter. If this parameter is not specified, all filters are returned.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterData

Gets filter that has the specified data.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FilterType

Gets filter of the specified type.

Type:	String
Accepted values:	AccessControl, BranchRepeater, ClientIP, ClientName, DesktopGroup, DesktopKind, DesktopTag, OU, User
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsAllowed

Gets allowed filters.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsEnabled

Gets enabled filters.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicyGuid

Gets filters that are defined for the policy.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoFilter

Get-BrokerGpoFilter returns an object for each matching GPO filter object.

Related Links

- about_Broker_GroupPolicy
- New-BrokerGpoFilter
- Set-BrokerGpoFilter
- Remove-BrokerGpoFilter

Get-BrokerGpoPolicy

March 11, 2024

Gets GPO policies configured for this site.

Syntax

```
Get-BrokerGpoPolicy
   [[-Name] <String>]
   [-Description <String>]
   [-IsEnabled <Boolean>]
   [-PolicySetGuid <Guid>]
   [-Priority <Int32>]
```



```
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerGpoPolicy
    [-PolicyGuid] <Guid>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve GPO policies matching the specified criteria. If no parameters are specified this cmdlet enumerates all GPO policies.

A GPO policy contains settings that control VDA behaviors.

————————BrokerGpoPolicy Object

A GPO policy object is a collection of VDA configurations that control how a VDA should behave. The filters in a policy are used to determine which VDAs are to be enforced with the specified configuration settings. Policy priorities determine how the policies should be applied together.

- Description (System.String)

  A short description.

- IsEnabled (System.Boolean)

  Specifies whether the policy is enabled. A disabled policy is not used to enforce its settings by VDAs.

- Name (System.String)

  The name of the policy object. Names for each policy object must be unique in a policy set.

- PolicyGuid (System.Guid)

  The GUID of the policy object.

- PolicySetGuid (System.Guid)

  The GUID of the policy set that contains this policy.

- Priority (System.Int32)

Specifies the priority of this policy among all the other policies in the policy set that contains this policy. The value is 1-based and the maximal value is the number of policies in the policy set.

Examples

EXAMPLE 1

Finds the policy named as 'Security settings'.

```
Get-BrokerGpoPolicy -Name 'Security settings'
```

EXAMPLE 2

Finds all policies in the default site policy set.

```
Get-BrokerGpoPolicy
```

Parameters

-PolicyGuid

Gets only the policy object with the specified GUID.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only the policy object with the specified name.

Type:	String
Position:	2

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets the policy that has the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsEnabled

Gets policies that are enabled or disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicySetGuid

Gets only the policies in the policy set.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Priority

Gets the policy that has the specified priority.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicy

Get-BrokerGpoPolicy returns an object for each matching policy.

Related Links

- about_Broker_GroupPolicy
- New-BrokerGpoPolicy
- Set-BrokerGpoPolicy
- Remove-BrokerGpoPolicy

Get-BrokerGpoPolicySet

March 11, 2024

Gets GPO policy sets defined in site.

Syntax

```
Get-BrokerGpoPolicySet
    [[-Name] <String>]
    [-Description <String>]
    [-IsAssigned <Boolean>]
    [-PolicyCount <Int32>]
    [-PolicySetType <String>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerGpoPolicySet
    [-PolicySetGuid] <Guid>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve information about all policy sets or a specified policy set.

———————BrokerGpoPolicySet Object

A GPO policy set is a container of GPO policies. There is always a site policy set, which contains all the site policies. There is also a policy set that contains all the policy templates. Other policy sets are private policy sets used by individual components for special configurations.

- Description (System.String)

  A short description of the policy set.

- IsAssigned (System.Boolean)

  True if the policy set is assigned to a delivery group.

- LastError (System.String)

  If neither nor empty, an error string indicating the reason why the most recent conversion from the policy set to VDA blob failed.

- Name (System.String)

  The name of the policy set. This must be unique among all policy sets.

- PolicyCount (System.Int32)

  Number of policies in the policy set.

- PolicySetGuid (System.Guid)

  The ID of the configuration policy set.

- PolicySetType (System.String)

  The policy set type.

- Scopes (Citrix.Fma.Sdk.DelegatedAdminInterfaces.ScopeReference[])

  The list of the delegated admin scopes to which the policy set belongs.

Examples

EXAMPLE 1

Lists the policy sets currently defined for the site.

```
1  Get-BrokerGpoPolicySet
```

Parameters

-PolicySetGuid

Gets specified policy set. If this parameter is not specified, all policy sets are returned.

Type:	Guid
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets specified policy set. If this parameter is not specified, all policy sets are returned.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets the policy set that has the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsAssigned

Gets policy sets that are currently assigned to at least one delivery group or not assigned to any delivery group.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicyCount

Gets the policy set that has the specified number of policies.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicySetType

Gets the policy sets of the specified type.

Type:	String
Accepted values:	CustomTemplates, DeliveryGroupPolicies, SitePolicies, SiteTemplates
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets the policy sets that are associated with the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets the policy sets that are associated with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicySet

Get-BrokerGpoPolicySet returns the matching policy set object.

Related Links

- about_Broker_GroupPolicy
- New-BrokerGpoPolicySet
- Set-BrokerGpoPolicySet
- Remove-BrokerGpoPolicySet

Get-BrokerGpoSetting

March 11, 2024

Gets GPO settings for a policy.

Syntax

```
Get-BrokerGpoSetting
    [[-SettingName] <String>]
    [-PolicyGuid <Guid>]
    [-SettingValue <String>]
    [-UseDefault <Boolean>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerGpoSetting
    [-SettingGuid] <Guid>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve GPO settings that meet certain criteria.

————————BrokerGpoSetting Object

A GPO setting specifies a configuration setting that controls how a VDA behaves.

- PolicyGuid (System.Guid)

  The GUID of the policy that owns the setting.

- SettingGuid (System.Guid)

  The GUID of the setting.

- SettingName (System.String)

  The name of the setting.

- SettingValue (System.String)

  The setting value.

- UseDefault (System.Boolean)

  Indicate if the default value of the setting is used. Ignored if setting is Boolean.

Examples

EXAMPLE 1

Gets the specified setting.

```
Get-BrokerGpoSetting -SettingGuid ([Guid]"1234abcd-...")
```

EXAMPLE 2

The value of the setting in the output may be produced if a setting with the value [“a”, “b”] has been created previously.

SettingValue : [“a”, “b”]

```
Get-BrokerGpoSetting -SettingValue '*"a"*"b"*'
```

Parameters

-SettingGuid

Gets specified setting. If this parameter is not specified, all settings that belong to the policy are returned.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SettingName

Gets specified setting if this parameter is specified.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PolicyGuid

Specifies the policy for which the settings are retrieved.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettingValue

Gets the setting with the specified value.

Getting one or more setting objects is a SQL search operation. The parameters are used as the conditions in a SQL WHERE clause. If the value of a setting contains characters that are used as SQL meta characters, for example, if the value of a setting is a JSON string [“a”, “b”], because the [,], and ‘‘are meta characters, searching using such a string may not yield the expected result. In general, it is not necessary to search by value when the GUID of a setting is known, or when the GUID of a policy and the name of a setting in the policy is known.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UseDefault

Gets the settings that use the default value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoSetting

Get-BrokerGpoSetting returns an object for each matching setting object.

Related Links

- about_Broker_GroupPolicy
- New-BrokerGpoSetting
- Set-BrokerGpoSetting
- Remove-BrokerGpoSetting

Get-BrokerHostingPowerAction

March 11, 2024

Gets power actions queued for machines.

Syntax

```
Get-BrokerHostingPowerAction
    [[-MachineName] <String>]
    [-Action <PowerManagementAction>]
    [-ActionCompletionTime <DateTime>]
    [-ActionStartTime <DateTime>]
    [-ActualPriority <Int32>]
    [-BasePriority <Int32>]
    [-DNSName <String>]
    [-FailureReason <String>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-HypHypervisorConnectionUid <Guid>]
    [-Metadata <String>]
    [-Origin <PowerActionPriority>]
    [-RequestTime <DateTime>]
    [-Sid <String>]
    [-State <PowerActionState>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerHostingPowerAction
    [-Uid] <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Finds power actions matching the specified search criteria from the queue of all known power actions. These power actions can be waiting to be dealt with or can be part way through being processed by the relevant hypervisor, or they can be recently completed actions. Completed actions are removed from the queue after a configured period, the default being one hour.

If no search criteria are specified all actions in the queue are obtained.

A Hosting Power Action record defines the action that is to be performed or has been performed, the machine that the action is to be applied to, the priority of the action in relation to other actions in the queue, times for points in the life of the action, and any results if the action has completed.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement' .

————————————BrokerHostingPowerAction Object

The BrokerHostingPowerAction object represents an instance of a power action. It contains the following properties:

- Action (Citrix.Broker.Admin.SDK.PowerManagementAction)

  The power action to be performed. Possible values are: TurnOn, TurnOff, Shutdown, Reset, Restart, Suspend, Resume.

- ActionCompletionTime (System.DateTime?)

  The time when the power action was completed by the hypervisor connection.

- ActionStartTime (System.DateTime?)

  The time when the power action was started to be processed by the hypervisor.

- ActualPriority (System.Int32)

  The current priority of the operation after any priority boosting.

- BasePriority (System.Int32)

  The starting priority of the operation.

- DNSName (System.String)

  The fully qualified DNS name of the machine that the power action applies to.

- FailureReason (System.String)

  For failed, lost, canceled or deleted actions, an indication of why the action did not complete.

- HostedMachineId (System.String)

  The hypervisor's ID for the machine that the power action applies to.

- HostedMachineName (System.String)

  The hypervisor's name for the machine that the power action applies to.

- HypervisorConnectionUid (System.Int32)

  The unique identifier of the hypervisor connection that is associated with the target machine.

- HypHypervisorConnectionUid (System.Guid)

  The UUID of the hypervisor connection that the machine's hosting server is accessed through. This is the identifier used by the Host and Provisioning services.

- MachineName (System.String)

  The name of the machine that the power action applies to, in the form domain\machine.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this power action.

- Origin (Citrix.Broker.Admin.SDK.PowerActionPriority)

  The origin of the power action.

- RequestTime (System.DateTime)

  The timestamp of when the action was created and placed in the queue.

- Sid (System.String)

  The SID of the machine to which the power action applies.

- State (Citrix.Broker.Admin.SDK.PowerActionState)

  The current state of this power action. Possible values are: Pending, Started, Completed, Failed, Canceled, Deleted, Lost.

- Uid (System.Int64)

  The unique identifier of the power action.

Examples

EXAMPLE 1

Fetches records for all known power actions either waiting to be processed, or currently being processed, or which have been processed in the last hour.

```
Get-BrokerHostingPowerAction
```

EXAMPLE 2

Fetches records for all power actions that are waiting to be processed and where the action is for a virtual machine that is hosted by the hypervisor called 'XenPool5'.

```
Get-BrokerHostingPowerAction -State Pending -HypervisorConnectionName '
    XenPool5'
```

Parameters

-Uid

Gets only the single action record whose ID matches the specified value.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets only the records for actions that are for machines whose name (of the form domain\machine) matches the specified string.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Action

Gets only action records with the specified action type.

Valid values are TurnOn, TurnOff, ShutDown, Reset, Restart, Suspend and Resume.

Type:	PowerManagementAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ActionCompletionTime

Gets only action records reported as having completed successfully at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ActionStartTime

Gets only action records reported as starting to be processed by the relevant hypervisor at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ActualPriority

Gets only the records for actions whose current active priority matches the specified value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-BasePriority

Gets only the records for actions whose original priority matches the specified value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets only the records for actions that are for machines whose DNS name matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FailureReason

Gets only actions that did not complete successfully and whose failure reason value matches the specified string.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineId

Gets only actions for machines whose hosting ID (the ID used by the hypervisor) matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets only actions for machines whose hosting name (the name used by the hypervisor) matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionName

Gets only the records for actions for machines hosted via a hypervisor connection whose name matches the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets only the records for actions for machines hosted via a hypervisor connection whose UID matches the specified value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypHypervisorConnectionUid

Gets only the records for actions for machines hosted via a hypervisor connection whose UUID matches the specified value.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Origin

Gets only actions whose origin matches the specified string.

Valid values are Reset, Schedule, Launch, Admin, Untaint, Policy and IdlePool.

Type:	PowerActionPriority
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RequestTime

Gets only the records for actions created and added to the queue at the specified time. This is useful with advanced filtering; for more information, see about_Broker_Filtering.

Type:	DateTime
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Gets only actions for machines whose SIDs match the specified string.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-State

Gets only the records for actions with the specified current state.

Valid values are Pending, Started, Completed, Failed, Canceled, Deleted and Lost.

Type:	PowerActionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.HostingPowerAction

Get-BrokerHostingPowerAction returns all power actions that match the specified selection criteria.

Related Links

- about_Broker_PowerManagement
- about_Broker_Filtering
- New-BrokerHostingPowerAction
- Set-BrokerHostingPowerAction
- Remove-BrokerHostingPowerAction

Get-BrokerHypervisorAlert

March 11, 2024

Gets hypervisor alerts recorded by the controller.

Syntax

```
Get-BrokerHypervisorAlert
   [-HostingServerName <String>]
   [-HypervisorConnectionUid <Int32>]
   [-Metadata <String>]
   [-Metric <AlertMetric>]
   [-Severity <AlertSeverity>]
   [-Time <DateTime>]
   [-TriggerInterval <TimeSpan>]
   [-TriggerLevel <Double>]
   [-TriggerPeriod <TimeSpan>]
   [-TriggerValue <Double>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Get-BrokerHypervisorAlert
   -Uid <Int64>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Get-BrokerHypervisorAlert cmdlet gets alert objects reported by the hypervisors that the controller is monitoring.

Without parameters, Get-BrokerHypervisorAlert gets all of the alerts recorded. Use parameters to select which alerts are returned.

Once you have configured suitable alerts in your hypervisor, and configured the controller with your hypervisor details (see New-BrokerHypervisorConnection), the controller monitors each hypervisor for new alerts.

Four hypervisor alert metrics are recorded; these relate to the hypervisor host, not individual virtual machines:

- Cpu: Reports excessive CPU usage.
- Memory: Reports excessive memory usage.
- Network: Reports high network activity.
- Disk: Reports high disk activity.

Each alert also includes information about where and when the alert occurred, the severity of the alert (Red/Yellow), and the configuration of the triggered alert.

The following metrics are supported with these hypervisors:

- VMware ESX (Cpu, Memory, Network, Disk)
- Citrix XenServer (Cpu, Network)
- Microsoft Hyper-V (None)

——————————BrokerHypervisorAlert Object

The BrokerHypervisorAlert represents a hypervisor alert object. It contains the following properties:

- HostingServerName (System.String)

  The name of the hypervisor hosting this machine.

- HypervisorConnectionUid (System.Int32)

  The Uid of the hypervisor connection that reported this alert.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this hypervisor alert.

- Metric (Citrix.Broker.Admin.SDK.AlertMetric)

  The metric this alert relates to: Cpu, Memory, Network, Disk, HostConnection or HostReboot.

- Severity (Citrix.Broker.Admin.SDK.AlertSeverity)

  Severity of the alert (Red or Yellow). Red is more serious than Yellow.

- Time (System.DateTime)

  Time that the alert occurred.

- TriggerInterval (System.TimeSpan?)

  Number of ticks (100ns) before the alert can be raised again.

- TriggerLevel (System.Double?)

  Threshold level that the alert was configured to trigger at.

- TriggerPeriod (System.TimeSpan?)

  Duration the value was above the trigger level.

- TriggerValue (System.Double?)

  The value of the monitored metric that triggered the alert.

- Uid (System.Int64)

  The unique internal identifier of this alert.

Examples

EXAMPLE 1

Returns all serious (Red) alerts for any hosting server with a name that starts with 'TestHyp'.

```
Get-BrokerHypervisorAlert -HostingServerName TestHyp* -Severity Red
```

EXAMPLE 2

Returns all CPU usage alerts that occurred in the last hour.

```
Get-BrokerHypervisorAlert -Filter {
  Metric -eq 'Cpu' -and Time -ge '-1:0'  }
```


Parameters

-Uid

Gets the hypervisor alert with the specified UID.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingServerName

Gets alerts for the specified hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets alerts for the specified hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metric

Gets alerts for a specified metric.

Valid values are: Cpu, Memory, Network and Disk.

Type:	AlertMetric
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Severity

Gets alerts with the specified severity.

Valid values are: Red and Yellow.

Type:	AlertSeverity
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Time

Gets alerts that occurred at a specific time.

You can also use -Filter and relative comparisons; see the examples for more information.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TriggerInterval

Gets alerts with a specific trigger interval. This is the interval before the alert is raised again.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TriggerLevel

Gets alerts with a specific trigger threshold level.

Type:	Double
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TriggerPeriod

Gets alerts with a specific trigger period. This is the duration the threshold level was exceeded for, prior to the alert triggering.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TriggerValue

Gets the value of the monitored metric that triggered the alert.

Type:	Double
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.HypervisorAlert

Get-BrokerHypervisorAlert returns an object for each matching alert.

Related Links

- about_Broker_Filtering
- New-BrokerHypervisorConnection

Get-BrokerHypervisorConnection

March 11, 2024

Gets hypervisor connections matching the specified criteria.

Syntax

```
Get-BrokerHypervisorConnection
   [[-Name] <String>]
   [-ExplicitPreferredController <Boolean>]
   [-ExtraSpinUpTimeSecs <Int32>]
   [-FaultReason <String>]
   [-FaultState <String>]
   [-FaultStateDuration <TimeSpan>]
   [-HypHypervisorConnectionUid <Guid>]
   [-HypHypervisorType <String>]
   [-IsReady <Boolean>]
   [-MachineCount <Int32>]
   [-MaxAbsoluteActiveActions <Int32>]
   [-MaxAbsoluteNewActionsPerMinute <Int32>]
   [-MaxAbsolutePvdPowerActions <Int32>]
   [-MaxPercentageActiveActions <Int32>]
   [-MaxPvdPowerActionsPercentageOfDesktops <Int32>]
   [-Metadata <String>]
```



```
    [-PreferredController <String>]
    [-State <HypervisorConnectionState>]
    [-TimeFaultStateEntered <DateTime>]
    [-ZoneExternalUid <Guid>]
    [-ZoneHealthy <Boolean>]
    [-ZoneUid <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerHypervisorConnection
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerHypervisorConnection cmdlet gets hypervisor connections matching the specified criteria. If no parameters are specified this cmdlet enumerates all hypervisor connections.

————————BrokerHypervisorConnection Object

The BrokerHypervisorConnection represents hypervisor connection object. It contains the following properties:

- Capabilities (System.String[])

  The set of capabilities as reported by the hypervisor.

- ExplicitPreferredController (System.Boolean?)

  Whether the PreferredController property was explicity set through the SDK or automatically chosen.

- ExtraSpinUpTimeSecs (System.Int32?)

  The extra time a VM is allowed to start before it is marked as failed on session launch

- FaultReason (System.String)

  Detailed fault state description if connection is currently in a fault state.

- FaultState (System.String)

Current fault state indicator associsted with hypervisor connection, or 'None'if connection is operating normally.

- FaultStateDuration (System.TimeSpan?)

  Period for which the hypervisor has been in fault state

- HypervisorCapabilities (System.String[])

  The set of hypervisor capabilities as reported by the hypervisor.

- HypHypervisorConnectionUid (System.Guid)

  The Guid that identifies the hypervisor connection.

- HypHypervisorType (System.String)

  The type of hypervisor connected to.

- IsReady (System.Boolean)

  Indicates that the connection is ready to be used in the configuration of managed machines.

- MachineCount (System.Int32)

  Count of machines associated with this hypervisor connection.

- MaxAbsoluteActiveActions (System.Int32?)

  Maximum number of active power actions allowed at any one time (defined in the metadata named 'Citrix_Broker_MaxAbsoluteActiveActions'on the hypervisor connection in the Citrix Hosting Service).

- MaxAbsoluteNewActionsPerMinute (System.Int32?)

  Maximum number of new actions that can be fired off to the hypervisor in any one minute (defined in the metadata named 'Citrix_Broker_MaxAbsoluteNewActionsPerMinute'on the hypervisor connection in the Citrix Hosting Service).

- MaxAbsolutePvdPowerActions (System.Int32?)

  This property is no longer supported.

- MaxPercentageActiveActions (System.Int32?)

  Maximum percentage of machines on the connection that can have active power actions at any one time (defined in the metadata named 'Citrix_Broker_MaxPowerActionsPercentageOfDesktops' on the hypervisor connection in the Citrix Hosting Service).

- MaxPvdPowerActionsPercentageOfDesktops (System.Int32?)

  This property is no longer supported.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Collection of all the metadata associated to the hypervisor connection.

- Name (System.String)

  The display name of the hypervisor connection.

- PreferredController (System.String)

  The name of the controller which is preferred to be used, when available, to perform all communication to the hypervisor. The name is in DOMAIN\machine form. A preferred controller may have been automatically chosen when the hypervisor connection was created.

- State (Citrix.Broker.Admin.SDK.HypervisorConnectionState)

  The state of the hypervisor connection.

- TimeFaultStateEntered (System.DateTime?)

  Time at which the hypervisor entered fault state

- Uid (System.Int32)

  Unique internal identifier of hypervisor connection.

- ZoneExternalUid (System.Guid?)

  The Guid that is the external zone uid of the hypervisor connection.

- ZoneHealthy (System.Boolean?)

  Health state of the Zone associated with this hypervisor connection.

- ZoneUid (System.Guid?)

  The Guid that identifies the zone associated with the hypervisor connection.

Examples

EXAMPLE 1

Gets a hypervisor connection by name.

```
1  $hvConn = Get-BrokerHypervisorConnection -Name "hvConnectionName"
```

EXAMPLE 2

Gets hypervisor connections by preferred controller.

```
1  $hvConn = Get-BrokerHypervisorConnection -PreferredController "
       domainName\controllerName"
```


EXAMPLE 3

Gets hypervisor connection used by a (power managed) machine.

```
$machine = Get-BrokerMachine -Uid $machineUid
$hvConn = Get-BrokerHypervisorConnection -Uid $machine.
    HypervisorConnectionUid
```

Parameters

-Uid

Gets the hypervisor connection with the specified internal id.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets hypervisor connections with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExplicitPreferredController

Gets hypervisor connections based on whether their preferred controller was explicitly specified or not

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExtraSpinUpTimeSecs

Gets the extra time a VM is allowed to start before it is marked as failed on session launch

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FaultReason

Gets hypervisor connections with fault reasons matching that specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FaultState

Gets hypervisor connections with fault states matching that specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FaultStateDuration

Period for which the hypervisor has been in fault state

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypHypervisorConnectionUid

Gets hypervisor connections with the specified Guid.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypHypervisorType

Gets hypervisor connections with the specified value of the hypervisor type.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsReady

Gets hypervisor connections with the specified value of the IsReady flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineCount

Gets hypervisor connections with the specified machine count.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAbsoluteActiveActions

Gets hypervisor connections with the specified MaxAbsoluteActiveActions value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAbsoluteNewActionsPerMinute

Gets hypervisor connections with the specified MaxAbsoluteNewActionsPerMinute value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAbsolutePvdPowerActions

This property is no longer supported.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPercentageActiveActions

Gets hypervisor connections with the specified MaxPercentageActiveActions value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPvdPowerActionsPercentageOfDesktops

This property is no longer supported.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredController

Gets hypervisor connections with the specified preferred controller. Specify the SAM name of the controller.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-State

Gets hypervisor connections with the specified connection state. Values can be can be:

- Unavailable - The broker is unable to contact the hypervisor.
- InMaintenanceMode - The hosting server is in maintenance mode.
- On - The broker is in contact with the hypervisor.

Type:	HypervisorConnectionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TimeFaultStateEntered

Time at which the hypervisor entered fault state

Type:	DateTime
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneExternalUid

Gets the hypervisor connections with the specified zone external uid.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneHealthy

Health state of the Zone associated with this hypervisor connection.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Gets the hypervisor connections with the specified zone uid.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.HypervisorConnection

Get-BrokerHypervisorConnection returns an object for each matching hypervisor connection.

Related Links

- New-BrokerHypervisorConnection
- Remove-BrokerHypervisorConnection
- Set-BrokerHypervisorConnection

Get-BrokerHypervisorConnectionStatus

March 11, 2024

Gets connection status and power management stats for the hypervisor.

Syntax

```
Get-BrokerHypervisorConnectionStatus
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

The GetBrokerHypervisorConnectionStatus cmdlet retrieves the power management statistics and connection status for all hypervisors on the DDC

————————————BrokerHypervisorConnectionStatus Object

Retrieves power management statistics and connection status for all hypervisors on the DDC

- CredentialsValid (System.Boolean?)

  Indicates whether the credentials are valid or not
- HclCurrentlyConnected (System.Boolean?)

  Indicates whether HCL is currently connected
- HypervisorConnectionGuid (System.Guid?)

  Gives the hypervisor connection GUID
- HypervisorConnectionUid (System.Int32)

  Gives the hypervisor connection UID
- HypervisorType (System.String)

  Type of hypervisor connection
- HypervisorZone (System.Guid?)

  Zone of the hypervisor connection
- InMaintenanceMode (System.Boolean?)

  Indicates whether the hypervisor is in maintenance mode or not
- InProgressActions (System.Int32?)

  Number of power actions in progress
- LastActionCompletedTime (System.DateTime?)

  Time when the last power action completed
- LastActionFailedTime (System.DateTime?)

  Time when the last power action failed
- LastActionStartTime (System.DateTime?)

  Time when the last power action started
- LastMachineManagerCreationTime (System.DateTime?)

  Time when the last machine manager was created

- LastPowerStateUpdateEventTime (System.DateTime?)

  Time when the last power state update event was created

- LastPowerStateUpdateTime (System.DateTime?)

  Time when the power state was last updated

- LastSiteServiceStartTime (System.DateTime?)

  Time when site services last started

- LastToolStateUpdateEventTime (System.DateTime?)

  Time when the last tool state update event was created

- LastToolStateUpdateTime (System.DateTime?)

  Time when the last tool state was updated

- NextPendingAction (System.Int64?)

  The next pending power action

- PendingActions (System.Int32?)

  Number of pending power actions

- TotalMachineManagersCreated (System.Int32?)

  Number of machine managers created

- TotalMachineManagersFailed (System.Int32?)

  Number of machine managers which failed

- TotalMachines (System.Int32?)

  Total number of machines present in the connection

Examples

Parameters

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GetBrokerHypervisorConnectionStatus

Get-BrokerHypervisorConnectionStatus returns an object for each hypervisor connection present in the DDC.

Related Links

Get-BrokerIcon

March 11, 2024

Get stored icons.

Syntax

```
Get-BrokerIcon
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerIcon
    -Uid <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerIcon
    -FileName <String>
    [-ServerName <String>]
    [-Index <Int32>]
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Reads a specific icon by Uid, or enumerates icons by passing no Uid.

————————BrokerIcon Object

The BrokerIcon object represents a single instance of an icon. It contains the following properties:

- EncodedIconData (System.String)

  The Base64 encoded .ICO format of the icon.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this command

- Uid (System.Int32)

  The UID of the icon itself.

Examples

EXAMPLE 1

Enumerate all icons.

```
Get-BrokerIcon
```

EXAMPLE 2

Get the icon with Uid 1.

```
Get-BrokerIcon -Uid 1
```

EXAMPLE 3

Retrieves the icon data from a file named “icon1.dll”on this computer. If this DLL contains multiple icons, all of them are returned in sequence.

```
Get-BrokerIcon -FileName icon1.dll
```

EXAMPLE 4

Retrieves only the first icon from the “c:\icons\icon1.dll”file on Server1.

```
Get-BrokerIcon -FileName c:\icons\icon1.dll -ServerName Server1 -Index
    0
```

EXAMPLE 5

Retrieves the icon data for an application named “app1.exe”, contained in an application profile named “app1.profile”located on a file share path “\\SERVER1\Share”. Returns only the first icon associated with that profiled application.

```
Get-BrokerIcon -FileName \\Server1\Share\app1.profile app1.exe -Index 0
```


Parameters

-Uid

Gets only the icon specified by unique identifier.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FileName

Specifies the name of a file from which to read the icon data. If the ServerName parameter is used, the FileName must be an absolute path.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

Specifies the name of the server. If the -FileName parameter refers to content or to a URL, the icon associated with the file type or URL is retrieved from the given server. Therefore, the server specified should have a file type handler installed for the given file type. If a local file path is specified, the server name refers to the server on which the file is located. If a UNC path is specified, the server name is unused.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Index

Specifies the zero-based icon resource index. For example, to select the first icon, specify an index of 0. Alternatively, to select the third icon, specify an index of 2. If the specified index is larger than the number of icons in the source file or profiled application, an error will be returned.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Icon

Returns an Icon object for each enumerated icon.

CtxIcon

Returns a list of zero or more icons, each in CtxIcon format. The CtxIcon.IconData property is the icon in standard Microsoft ICO format.

Related Links

- about_Broker_Filtering
- New-BrokerIcon
- Remove-BrokerIcon

Get-BrokerImportedFTA

March 11, 2024

Gets the imported file type associations.

Syntax

```
Get-BrokerImportedFTA
   [[-ExtensionName] <String>]
   [-ContentType <String>]
   [-Description <String>]
   [-DesktopGroupUid <Int32>]
   [-Edit <String>]
   [-EditArguments <String>]
   [-EditExecutableName <String>]
   [-HandlerName <String>]
   [-Open <String>]
   [-OpenArguments <String>]
   [-OpenExecutableName <String>]
   [-PerceivedType <String>]
   [-Print <String>]
   [-PrintTo <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
```



```
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerImportedFTA
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns the file type associations the system imports from worker machines.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Initially the system is not aware of any extensions, and they must be imported by the Citrix administrator. See the Update-BrokerImportedFTA cmdlet for more information.

After file type extensions are imported, this cmdlet lets the administrator review which file type associations the system is aware of. ImportedFTA objects are also used when configuring content redirection. See the New-BrokerConfiguredFTA cmdlet for more information.

The imported file type associations are grouped according to the desktop group to which they belong, because the system expects all machines in the same desktop group to have the same file type associations. That may not be true, however, across desktop groups.

Note that the ImportedFTA object has several fields that are not currently used. Only those fields that are shared with the ConfiguredFTA object are actually used in some capacity.

——————————BrokerImportedFTA Object

The BrokerImportedFTA object represents a file type association imported from worker machines. It contains the following properties:

- ContentType (System.String)

Content type of the file, such as “text/plain”or “application/vnd.ms-excel”.

- Description (System.String)

  File type description, such as “Test Document”, “Microsoft Word Text Document”, etc.

- DesktopGroupUid (System.Int32)

  The desktop group this file type belongs to.

- Edit (System.String)

  Edit command with full path to executable: “C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE”
  /n /dde

- EditArguments (System.String)

  The arguments used for the ‘edit’action on files of this type. These are extracted from the full edit command, and may be empty.

- EditExecutableName (System.String)

  The executable name extracted from the Edit property, no path included. This is used for matching with published apps’executable when searching for the list of extensions an application is capable of handling.

- ExtensionName (System.String)

  A single file extension, such as .txt. Unique within the scope of a desktop group.

- HandlerName (System.String)

  File type handler name, e.g. “Word.Document.8”or TXTFILE.

- Open (System.String)

  Open command with full path to executable: “C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EX
  /n /dde

- OpenArguments (System.String)

  The arguments used for the ‘open’action on files of this type. These are extracted from the full open command, and may be empty.

- OpenExecutableName (System.String)

  The executable name extracted from the Open property, no path included. This is used for matching with published apps’executable when searching for the list of extensions an application is capable of handling.

- PerceivedType (System.String)

  Perceived type, such as “text”.

- Print (System.String)

  Print command: “C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE”/x /n /dde

- PrintTo (System.String)

  PrintTo command: “C:\Program Files (x86)\Microsoft Office\Office12\WINWORD.EXE”/n /dde

- Uid (System.Int32)

  Unique internal identifier of imported file type association.

Examples

EXAMPLE 1

Invoking this cmdlet with no arguments simply returns
all of the imported file type association objects. By default, only
the first 250 objects are returned. See the -MaxRecordCount and
-Skip parameters for information about modifying this.

```
Get-BrokerImportedFTA
```

EXAMPLE 2

Retrieves all imported file type associations that have
the extension “.txt”. Note that because imported file type associations
are per-desktop group, multiple instances may be returned.

```
Get-BrokerImportedFTA -ExtensionName ".txt"
```

Parameters

-Uid

Gets only the imported file type associations with the
specified unique identifier.

Type:	Int32
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ExtensionName

Gets only the imported file type associations with the specified extension name. For example, “.txt”or “.png”.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ContentType

Gets only the imported file type associations with the specified content type (as listed in the Registry). For example, “application/msword”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets only the imported file type associations with the specified description (as listed in the Registry). For example, “Text Document”or “Microsoft Word text document”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only the file type associations imported from a worker machine belonging to the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Edit

Gets only the imported file type associations with the specified Edit command, that includes both the executable name and path, and any arguments to that executable.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-EditArguments

Gets only the imported file type associations with the specified arguments to the Edit command.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-EditExecutableName

Gets only the imported file type associations with the specified executable for the Edit command.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HandlerName

Gets only the imported file type associations with the specified handler name (as listed in the Registry). For example, "TXTFILE"or "Word.Document.8".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Open

Gets only the imported file type associations with the specified Open command, that includes both the executable name and path, and any arguments to that executable.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OpenArguments

Gets only the imported file type associations with the specified arguments to the Open command.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OpenExecutableName

Gets only the imported file type associations with the specified executable for the Open command.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PerceivedType

Gets only the imported file type associations with the specified perceived type (as listed in the Registry). For example, “document”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Print

Gets only the imported file type associations with the

specified Print command.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PrintTo

Gets only the imported file type associations with the

specified PrintTo command.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

This cmdlet does not accept any input from the pipeline.

Outputs

Citrix.Broker.Admin.SDK.ImportedFTA

One or more ImportedFTA objects are returned as output.

Related Links

- about_Broker_Applications
- New-BrokerConfiguredFTA
- Remove-BrokerImportedFTA
- Update-BrokerImportedFTA

Get-BrokerInstalledDbVersion

March 11, 2024

Gets a list of all available database schema versions for the Broker Service.

Syntax

```
Get-BrokerInstalledDbVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Broker Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-BrokerDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Broker Service database schema version number.

```
Get-BrokerInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Broker Service database schema for which upgrade scripts are supplied.

```
Get-BrokerInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-BrokerInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Broker

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerDBVersionChangeScript
- Get-BrokerDBSchema

Get-BrokerLease

March 11, 2024

Gets stored leases.

Syntax

```
Get-BrokerLease
    [[-Key] <String>]
    [-Expiration <DateTime>]
    [-LastModified <DateTime>]
    [-LeaseType <BrokerLeaseType>]
    [-OwnerSAMName <String>]
    [-OwnerSID <String>]
    [-OwnerUPN <String>]
    [-ZoneUid <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerLease
    [-Uid] <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets leases filtered by specific Uid or Owner information.

——————————BrokerLease Object

The BrokerLease object represents a single instance of a lease. It contains the following properties:

- Expiration (System.DateTime)

  The expiration time of the lease.

- Key (System.String)

  The SHA1 representation of the lease key.

- LastModified (System.DateTime)

  The modification time of the lease.

- LeaseType (Citrix.Broker.Admin.SDK.BrokerLeaseType)

  The type of lease.

- OwnerSAMName (System.String)

  The SAM name of the user associated with the lease.

- OwnerSID (System.String)

  The SID of the user associated with the lease.

- OwnerUPN (System.String)

  The UPN of the user associated with the lease.

- Uid (System.Int64)

  The UID of the lease itself.

- Value (System.String)

  The serialized lease data in XML.

- ZoneName (System.String)

  Name of the Zone this resource belongs to.

- ZoneUid (System.Guid?)

  Uid of the Zone this resource belongs to.

Examples

EXAMPLE 1

Gets the lease with internal Uid 1.

```
$lease = Get-BrokerLease -Uid 1
```

EXAMPLE 2

Gets the leases associated with the specified user.

```
$leases = Get-BrokerLease -OwnerSAMName Domain\User
```

Parameters

-Uid

Gets only the lease specified by unique identifier.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Key

Gets only the leases matching the specified lease key pattern.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Expiration

Gets only the leases matching the specified expiration date and time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastModified

Gets only the leases matching the specified modified date and time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeaseType

Gets only leases of a specific type. Possible values Enumeration, Launch.

Type:	BrokerLeaseType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OwnerSAMName

Gets only the leases associated with the specified Domain\User.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OwnerSID

Gets only the leases associated with the specified user SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OwnerUPN

Gets only the leases associated with the specified user UPN.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets only the leases of resources beloning to Zone with specified Uid.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Lease

Returns an Lease object for each enumerated lease.

Related Links

- Remove-BrokerLease
- Update-BrokerLocalLeaseCache

Get-BrokerMachine

March 11, 2024

Gets machines belonging to this site.

Syntax

```
Get-BrokerMachine
    [[-MachineName] <String>]
    [-AgentVersion <String>]
    [-AllocationType <AllocationType>]
    [-ApplicationInUse <String>]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-AssignedUserSID <String>]
    [-AssociatedTenantId <Guid>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserSID <String>]
    [-AssociatedUserUPN <String>]
    [-AzureADJoinedMode <String>]
    [-AzureDeviceId <String>]
    [-BrowserName <String>]
    [-CatalogName <String>]
```



```
    [-CatalogUid <Int32>]
    [-CatalogUUID <Guid>]
    [-CbpVersion <CBPVersion>]
    [-ColorDepth <ColorDepth>]
    [-ControllerDNSName <String>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DesktopCondition <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-DesktopGroupUUID <Guid>]
    [-DesktopKind <DesktopKind>]
    [-DesktopUid <Int32>]
    [-DNSName <String>]
    [-DrainingUntilShutdown <Boolean>]
    [-FaultState <MachineFaultState>]
    [-FunctionalLevel <FunctionalLevel>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-HypHypervisorConnectionUid <Guid>]
    [-IconUid <Int32>]
    [-ImageOutOfDate <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAssigned <Boolean>]
    [-IsPhysical <Boolean>]
    [-IsReserved <Boolean>]
    [-LastAssignmentTime <DateTime>]
    [-LastConnectionFailure <ConnectionFailureReason>]
    [-LastConnectionTime <DateTime>]
    [-LastConnectionUser <String>]
    [-LastDeregistrationReason <DeregistrationReason>]
    [-LastDeregistrationTime <DateTime>]
    [-LastErrorReason <String>]
    [-LastErrorTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-LastPvdErrorReason <String>]
    [-LastPvdErrorTime <DateTime>]
    [-LastRegistrationTime <DateTime>]
    [-LoadIndex <Int32>]
    [-MacAddress <String>]
    [-MachineInternalState <MachineInternalState>]
    [-MachineUnavailableReason <String>]
    [-MaintenanceModeReason <MaintenanceModeReason>]
    [-Metadata <String>]
    [-NameLookupFailureCount <Int32>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-PersistUserChanges <PersistUserChanges>]
    [-PowerActionPending <Boolean>]
```



```
    [-PowerState <PowerState>]
    [-ProvisioningType <ProvisioningType>]
    [-PublishedApplication <String>]
    [-PublishedName <String>]
    [-PvdEstimatedCompletionTime <DateTime>]
    [-PvdPercentDone <Int32>]
    [-PvdStage <PvdStage>]
    [-PvdUpdateStartTime <DateTime>]
    [-RegistrationState <RegistrationState>]
    [-ScheduledReboot <ScheduledReboot>]
    [-SecureIcaRequired <Boolean>]
    [-SessionAutonomouslyBrokered <Boolean>]
    [-SessionClientAddress <String>]
    [-SessionClientName <String>]
    [-SessionClientVersion <String>]
    [-SessionConnectedViaHostName <String>]
    [-SessionConnectedViaIP <String>]
    [-SessionCount <Int32>]
    [-SessionDeviceId <String>]
    [-SessionHardwareId <String>]
    [-SessionHidden <Boolean>]
    [-SessionKey <Guid>]
    [-SessionLaunchedViaHostName <String>]
    [-SessionLaunchedViaIP <String>]
    [-SessionProtocol <String>]
    [-SessionSecureIcaActive <Boolean>]
    [-SessionsEstablished <Int32>]
    [-SessionSmartAccessTag <String>]
    [-SessionsPending <Int32>]
    [-SessionStartTime <DateTime>]
    [-SessionState <SessionState>]
    [-SessionStateChangeTime <DateTime>]
    [-SessionSupport <SessionSupport>]
    [-SessionType <SessionType>]
    [-SessionUid <Int64>]
    [-SessionUserName <String>]
    [-SessionUserSID <String>]
    [-SID <String>]
    [-SummaryState <DesktopSummaryState>]
    [-SupportedPowerAction <String>]
    [-Tag <String>]
    [-UUID <Guid>]
    [-VMToolsState <VMToolsState>]
    [-WillShutdownAfterUse <Boolean>]
    [-WillShutdownAfterUseReason <WillShutdownAfterUseReason>]
    [-WindowsConnectionSetting <WindowsConnectionSetting>]
    [-ZoneHealthy <Boolean>]
    [-ZoneName <String>]
    [-ZoneUid <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
```



```
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerMachine
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves machines matching the specified criteria. If no parameters are specified, this cmdlet enumerates all machines.

Get-BrokerMachine returns objects that combine machine configuration and state information.

For single-session machines, session information is displayed if present. If "fast user switching"is enabled, more than one session may be present on single-session machines. Because this cmdlet returns information only for a single session, if two sessions are present it will return information about the brokered session (rather than, for example, an unbrokered direct RDP session). If there is no session running, session-related fields return $null.

For multi-session machines, no session information about single sessions is displayed by this cmdlet, and so are always $null. Get-BrokerSession can be used to get information about sessions on both multi-session and single-session machines.

To count machines, rather than retrieve full details of each machine, use Group-BrokerMachine instead.

See about_Broker_Filtering for information about advanced filtering options, and about_Broker_Machines for background information about machines.

——————————BrokerMachine Object

The machine object returned represents a physical or virtual machine, which has been configured in the site.

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the machine.

- AllocationType (Citrix.Broker.Admin.SDK.AllocationType)

  Describes how the machine is allocated to the user, can be Random or Static.

- ApplicationsInUse (System.String[])

  List of applications in use on the machine (in the form of browser name).

- AssignedClientName (System.String)

  The name of the endpoint client device that the machine has been assigned to.

- AssignedIPAddress (System.String)

  The IP address of the endpoint client device that the machine has been assigned to.

- AssignedUserSIDs (System.String[])

  The SIDs of the users that have been assigned to the machine (private machines only).

- AssociatedTenantId (System.Guid?)

  Tenant associated with the machine. Once a tenant is associated with a machine it can only host sessions for that tenant and no other. Tenant associations are transient and are cleared when the machine's image is reset.

- AssociatedUserFullNames (System.String[])

  Full names of the users that have been associated with the machine (usually in the form "First-name Lastname").
  Associated users are the current user(s) for shared machines and the assigned users for private machines.

- AssociatedUserNames (System.String[])

  Usernames of the users that have been associated with the machine (in the form "domain\user").
  Associated users are the current user(s) for shared machines and the assigned users for private machines.

- AssociatedUserSIDs (System.String[])

  The SIDs of the users that have been associated with the machine.
  Associated users are the current user(s) for shared machines and the assigned users for private machines.

- AssociatedUserUPNs (System.String[])

  The User Principal Names of the users associated with the machine (in the form user@domain).
  Associated users are the current user(s) for shared machines and the assigned users for private machines.

- AzureADJoinedMode (System.String)

  The type of Azure AD Domain join that took place in VDA.
  This will be NotAadJoined when the VDA is not Azure AD domain joined.

- AzureDeviceId (System.String)

  The Azure DeviceId for this worker. Available if the worker is
  is registered with AzureAD.

- BrowserName (System.String)

  Site-wide unique name identifying associated desktop to other components (for example StoreFront). This is typically non-null only for machines backing assigned private desktops.

- Capabilities (System.String[])

  List of the capabilities that the machine supports. Valid capabilities are:

  - MultiSession: Indicates an RDS- (Terminal Services-) based machine, which supports multiple active sessions from different users.
  - CBP1_5: Indicates the machine can use the CBP 1.5 protocol for communication.

- CatalogName (System.String)

  Name of the catalog the machine is a member of.

- CatalogUid (System.Int32)

  UID of the catalog the machine is a member of.

- CatalogUUID (System.Guid)

  UUID of the catalog the machine is a member of.

- CbpVersion (Citrix.Broker.Admin.SDK.CBPVersion?)

  The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

  The color depth setting configured on the machine, possible values are:
  $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

- ControllerDNSName (System.String)

  The DNS host name of the controller that the machine is registered to.

- DeliveryType (Citrix.Broker.Admin.SDK.DeliveryType?)

  Denotes whether the machine delivers desktops only, apps only or both.

- Description (System.String)

  Description of the machine.

- DesktopConditions (System.String[])

  List of outstanding desktop conditions for the machine.

- DesktopGroupName (System.String)

  Name of the desktop group the machine is a member of.

- DesktopGroupUid (System.Int32?)

  UID of the desktop group the machine is a member of.

- DesktopGroupUUID (System.Guid?)

  UUID of the desktop group the machine is a member of.

- DesktopKind (Citrix.Broker.Admin.SDK.DesktopKind?)

  Deprecated.
  Denotes whether the machine is private or shared. Use AllocationType instead.

- DesktopUid (System.Int32?)

  The UID of the associated desktop object.

- DNSName (System.String)

  The DNS host name of the machine.

- DrainingUntilShutdown (System.Boolean)

  Flag indicating if this machine is draining and will be shut down after all sessions on the machine have ended. This flag is only ever true on power-managed, multi-session machines.
  Note: The machine will not shut down if it is in maintenance mode; it will shut down only after it is taken out of maintenance mode.

- FaultState (Citrix.Broker.Admin.SDK.MachineFaultState)

  Summary state of any current fault state of the machine. Can be one of the following:

  - None - No fault; machine is healthy.
  - FailedToStart - Last power-on operation for machine failed.
  - StuckOnBoot - Machine does not seem to have booted following power on.
  - Unregistered - Machine has failed to register within expected period, or its registration has been rejected.
  - MaxCapacity - Machine is reporting itself at maximum capacity.

- FunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel?)

  Functional level of the machine, if known.

- HostedMachineId (System.String)

  Unique ID within the hosting unit of the target managed machine.

- HostedMachineName (System.String)

The friendly name of a hosted machine as used by its hypervisor. This is not necessarily the DNS name of the machine.

- HostingServerName (System.String)

  DNS name of the hypervisor that is hosting the machine if managed.

- HypervisorConnectionName (System.String)

  The name of the hypervisor connection that the machine has been assigned to, if managed.

- HypervisorConnectionUid (System.Int32?)

  The UID of the hypervisor connection that the machine's hosting server is accessed through.

- HypHypervisorConnectionUid (System.Guid?)

  The UUID of the hypervisor connection that the machine's hosting server is accessed through

- IconUid (System.Int32?)

  The UID of the machine's icon that is displayed in StoreFront.

- ImageOutOfDate (System.Boolean?)

  Denotes if the VM image for a hosted machine is out of date.

- InMaintenanceMode (System.Boolean)

  Denotes if the machine is in maintenance mode.

- IPAddress (System.String)

  The IP address of the machine.

- IsAssigned (System.Boolean)

  Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explicitly or by assigning on first use of the machine.

- IsPhysical (System.Boolean)

  This value is true if the machine is physical (ie not power managed by the Citrix Broker service, and false otherwise.

- IsReserved (System.Boolean)

  Indicates if machine is reserved for special use, for example for AppDisk preparation. A reserved machine cannot be a member of a desktop group.

- LastAssignmentTime (System.DateTime?)

  The Datetime of the most recent assignment to a user, client IP address or client name.

- LastConnectionFailure (Citrix.Broker.Admin.SDK.ConnectionFailureReason)

  The reason for the last failed connection between a client and the machine.

- LastConnectionTime (System.DateTime?)

  Time of the last detected connection attempt that either failed or succeeded.

- LastConnectionUser (System.String)

  The SAM name (in the form DOMAIN\user) of the user that last attempted a connection with the machine. If the SAM name is not available, the SID is used.

- LastDeregistrationReason (Citrix.Broker.Admin.SDK.DeregistrationReason?)

  The reason for the last deregistration of the machine with the broker. Possible values are: AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- LastDeregistrationTime (System.DateTime?)

  Time of the last deregistration of the machine from the controller.

- LastErrorReason (System.String)

  The reason for the last error detected in the machine.

- LastErrorTime (System.DateTime?)

  The time of the last detected error.

- LastHostingUpdateTime (System.DateTime?)

  Time of last update to any hosting data (such as power states) or the last completed power action for this machine reported by the hypervisor connection.

- LastPvdErrorReason (System.String)

  This property is no longer supported.

- LastPvdErrorTime (System.DateTime?)

  This property is no longer supported.

- LastRegistrationTime (System.DateTime?)

  Time of last registration of the machine with the controller.

- LoadIndex (System.Int32?)

  Gives current effective load index for multi-session machines.

- LoadIndexes (System.String[])

  Gives the last reported individual load indexes that were used in the calculation of the LoadIndex value. Note that the LoadIndex value may have been subsequently adjusted due to session brokering operations. This value is only set for multi-session machines.

- MacAddress (System.String)

  A string representing the physical address of the network interface corresponding to the IP Address this machine uses to communicate with the Broker.

- MachineInternalState (Citrix.Broker.Admin.SDK.MachineInternalState)

  The internal state of the machine; reported while the machine is registered to a controller, plus some private Citrix Broker Service states while the machine is not registered.

- MachineName (System.String)

  DNS host name of the machine.

- MachineUnavailableReason (System.String)

  Reason reported from the VDA showing why it is not yet available for use even though it is registered

- MaintenanceModeReason (Citrix.Broker.Admin.SDK.MaintenanceModeReason)

  The reason why the machine was placed in maintenance mode (if it is in maintenance mode). Possible values are:

  - None - Machine is not in maintenance mode.
  - Administrator - Machine was manually placed in maintenance mode by an administrator.
  - MaxFailedRegistrations - Machine was automatically placed in maintenance mode due to reaching the maximum failed registration limit.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Any metadata that is associated with the machine.

- NameLookupFailureCount (System.Int32)

  Tracks the number of consecutive directory lookup failures for this account

- OSType (System.String)

  A string that can be used to identify the operating system that is running on the machine.

- OSVersion (System.String)

  A string that can be used to identify the version of the operating system running on the machine, if known.

- PersistUserChanges (Citrix.Broker.Admin.SDK.PersistUserChanges)

  Describes if and how user changes are persisted. Possible values are:

  - OnLocal - Persist the user changes on the local disk of the machine.
  - Discard - Discard user changes.

- PowerActionPending (System.Boolean)

  Indicates if there are any pending power actions for the machine.

- PowerState (Citrix.Broker.Admin.SDK.PowerState)

  The current power state of the machine. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, resuming.

- ProvisioningType (Citrix.Broker.Admin.SDK.ProvisioningType)

  Describes how the machine was provisioned, possible values are:

  - Manual: No automated provisioning.
  - PVS: Machine provisioned by PVS (may be physical, blade, VM,...)
  - MCS: Machine provisioned by MCS (machine must be VM)

- PublishedApplications (System.String[])

  List of applications published by the machine (displayed as browser names).

- PublishedName (System.String)

  The name of the machine that is displayed in StoreFront, if the machine has been published.

- PvdEstimatedCompletionTime (System.DateTime?)

  This property is no longer supported.

- PvdPercentDone (System.Int32?)

  This property is no longer supported.

- PvdStage (Citrix.Broker.Admin.SDK.PvdStage)

  This property is no longer supported.

- PvdUpdateStartTime (System.DateTime?)

  This property is no longer supported.

- RegistrationState (Citrix.Broker.Admin.SDK.RegistrationState)

  Indicates the registration state of the machine. Possible values are: Unregistered, Initializing, Registered, AgentError.

- ScheduledReboot (Citrix.Broker.Admin.SDK.ScheduledReboot)

  Indicates the state of any scheduled reboot operation for a machine. Possible values:

  - None: No reboot is scheduled.
  - Pending: Machine is awaiting reboot but is available for use.
  - Draining: Machine is awaiting reboot and is unavailable for new sessions; reconnections to existing connections are still allowed, however.
  - InProgress: Machine is actively undergoing a scheduled reboot. o Natural: Natural reboot in progress. Machine is awaiting a restart.

- SecureIcaRequired (System.Boolean?)

  Flag indicating whether SecureICA is required or not when starting a session on the machine.

- SessionAutonomouslyBrokered (System.Boolean?)

  Session property indicating if the current session is an HDX session established by direct connection without being brokered.
  Session properties are always null for multi-session machines.

- SessionClientAddress (System.String)

  Session property indicating the IP address of the client connected to the machine.
  Session properties are always null for multi-session machines.

- SessionClientName (System.String)

  Session property indicating the host name of the client connected to the machine.
  Session properties are always null for multi-session machines.

- SessionClientVersion (System.String)

  Session property indicating the version of the Citrix Receiver on the connected client.
  Session properties are always null for multi-session machines.

- SessionConnectedViaHostName (System.String)

  Session property indicating the host name of the connection gateway, router or client.
  Session properties are always null for multi-session machines.

- SessionConnectedViaIP (System.String)

  Session property indicating the IP address of the connection gateway, router or client.
  Session properties are always null for multi-session machines.

- SessionCount (System.Int32)

  Count of number of sessions on the machine.

- SessionDeviceId (System.String)

  Session property indicating a unique identifier for the client device that has most recently been associated with the current session.
  Session properties are always null for multi-session machines.

- SessionHardwareId (System.String)

  Session property indicating a unique identifier for the client hardware that has been most recently associated with the current session.
  Session properties are always null for multi-session machines.

- SessionHidden (System.Boolean?)

  Session property that indicates if a session is hidden.
  Session properties are always null for multi-session machines.

- SessionKey (System.Guid?)

  Session property indicating the key of the current session.
  Session properties are always null for multi-session machines.

- SessionLaunchedViaHostName (System.String)

  Session property that denotes the host name of the StoreFront server used to launch the current brokered session.
  Session properties are always null for multi-session machines.

- SessionLaunchedViaIP (System.String)

  Session property that denotes the IP address of the StoreFront server used to launch the current brokered session.
  Session properties are always null for multi-session machines.

- SessionProtocol (System.String)

  Session property that denotes the protocol that the current session is using, can be either HDX, RDP or Console. Console sessions on XenDesktop 5 VDAs appear with a blank protocol.
  Session properties are always null for multi-session machines.

- SessionSecureIcaActive (System.Boolean?)

  Session property that indicates whether SecureICA is active on the current session or not.
  Session properties are always null for multi-session machines.

- SessionsEstablished (System.Int32)

  Number of established sessions on this machine. For multi-session machines this excludes established sessions which have not yet completed their logon processing.

- SessionSmartAccessTags (System.String[])

  Session property that indicates the Smart Access tags for the current session.
  Session properties are always null on multi-session machines.

- SessionsPending (System.Int32)

Number of pending (brokered but not yet established) sessions on this machine. For multi-session machines this also includes established sessions which have not yet completed their logon processing.

- SessionStartTime (System.DateTime?)

  Session property that indicates the start time of the current session.
  Session properties are always null on multi-session machines.

- SessionState (Citrix.Broker.Admin.SDK.SessionState?)

  Session property indicating the state of the current session, possible values are:
  Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession and Unknown. Session properties are always null for multi-session machines.

- SessionStateChangeTime (System.DateTime?)

  Session property indicating the time of the last state change of the current session.
  Session properties are always null for multi-session machines.

- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport)

  Indicates the session support of the machine.
  Possible values:

  - SingleSession: Single-session only machine.
  - MultiSession: Multi-session capable machine.

- SessionType (Citrix.Broker.Admin.SDK.SessionType?)

  Session property indicating the type of the current session.
  Session properties are always null for multi-session machines.

- SessionUid (System.Int64?)

  Session property indicating the UID of the current session.
  Session properties are always null for multi-session machines.

- SessionUserName (System.String)

  Session property indicates the name of the current session's user (in the form DOMAIN\user).
  Session properties are always null for multi-session machines.

- SessionUserSID (System.String)

  Session property indicates the SID of the current session's user.
  Session properties are always null for multi-session machines.

- SID (System.String)

  The SID of the machine.

- SummaryState (Citrix.Broker.Admin.SDK.DesktopSummaryState)

  Indicates the overall state of the desktop associated with the machine. The overall state is a result of other more specific states such as session state, registration state and power state. Possible values: Off, Unregistered, Available, Disconnected, InUse, Preparing.

- SupportedPowerActions (System.String[])

  A list of power actions supported by this machine.

- Tags (System.String[])

  A list of tags associated with the machine.

- Uid (System.Int32)

  UID of the machine object.

- UUID (System.Guid)

  UUID of the machine object.

- VMToolsState (Citrix.Broker.Admin.SDK.VMToolsState)

  State of the hypervisor tools present on the VM (if any).
  Possible values are:
  NotPresent, Unknown, NotStarted, Running.

- WillShutdownAfterUse (System.Boolean)

  Flag indicating if this machine is tainted and will be shut down after all sessions on the machine have ended. This flag is only ever true on power-managed, single-session machines.
  Note: The machine will not shut down if it is in maintenance mode; it will shut down only after it is taken out of maintenance mode.

- WillShutdownAfterUseReason (Citrix.Broker.Admin.SDK.WillShutdownAfterUseReason)

  The reason why the machine will shutdown after use (if it will shutdown after use). Possible values are:

  - None - Machine will not shutdown after use.
  - ResetDiskImage - Machine will shutdown after use to reset its disk image.
  - ScheduledNaturalReboot - Machine will shutdown after use as part of the scheduled natural reboot process.
  - OnDemandNaturalReboot - Machine will shutdown after use as part of an on-demand natural reboot process.

- WindowsConnectionSetting (Citrix.Broker.Admin.SDK.WindowsConnectionSetting?)

  The logon mode reported by Windows itself (multi-session machines only). For single-session machines the value is always hardwired to LogonEnabled.

Possible values are:
LogonEnabled, Draining, DrainingUntilRestart and LogonDisabled.

- ZoneHealthy (System.Boolean?)

  The health state of the zone in which the machine is located.

- ZoneName (System.String)

  The name of the zone in which the machine is located.

- ZoneUid (System.Guid)

  The UID of the zone in which the machine is located.

Examples

EXAMPLE 1

These commands return all suspended machines. The second form uses advanced filtering (see about_Broker_Filtering).

```
Get-BrokerMachine -PowerState Suspended
Get-BrokerMachine -Filter {
  PowerState -eq 'Suspended'  }
```

EXAMPLE 2

This command returns all machines belonging to the DNS domain mydomain.mycompany.com.

```
Get-BrokerMachine -DNSName '*.mydomain.mycompany.com'
```

EXAMPLE 3

This command returns all registered machines running on the specified hypervisor connection.

```
Get-BrokerMachine -Filter {
  RegistrationState -eq 'Registered' -and HypervisorConnectionUid -eq 5
       }
```

EXAMPLE 4

This command finds all of the machines in MyDomain with names beginning with X and removes them from the specified desktop group.


```
Get-BrokerMachine -MachineName 'MyDomain\X*' | Remove-
    BrokerDesktopGroup 2
```

EXAMPLE 5

This command gets all desktops in a site. Use this instead of the deprecated Get-BrokerDesktop command.

```
Get-BrokerMachine -Filter {
  DesktopGroupUid -ne $null  }
```

Parameters

-Uid

Gets a machine with a specific UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets machines with a specific machine name (in the form domain\machine).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AgentVersion

Gets machines with a specific Citrix Virtual Delivery Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AllocationType

Gets machines from catalogs with the specified allocation type.

Type:	AllocationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationInUse

Gets machines running a specified published application (identified by browser name).

String comparisons are case-insensitive.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedClientName

Gets machines that have been assigned to the specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedIPAddress

Gets machines that have been assigned to the specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedUserSID

Gets machines with the specific SID of the user to whom the desktop is assigned.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-AssociatedTenantId

Gets machines associated with the specified tenant.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserFullName

Gets machines with an associated user identified by their full name (usually 'first-name last-name').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets machines with an associated user identified by their user name (in the form 'domain\user').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserSID

Gets machines with an associated user identified by their Windows SID.

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserUPN

Gets machines with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AzureADJoinedMode

Gets machines with a specific Azure AD Domain Join Type

- NotAadJoined - Machine not joined to Azure AD yet.
- HybridAadJoined - Machine was Hybrid Aad joined.
- PureAadJoined - Machine was Pure Aad joined.

Type:	String
Accepted values:	HybridAadJoined, NotAadJoined, PureAadJoined
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDeviceId

Gets machines with matching Azure DeviceId.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrowserName

Gets assigned machines backing desktop resources that have browser names matching the specified name.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogName

Gets machines from the catalog with the specific name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogUid

Gets machines from the catalog with the specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUUID

Gets machines from the catalog with the specific UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CbpVersion

The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

Type:	CBPVersion
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Gets machines configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets machines with a specific DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DeliveryType

Gets machines of a particular delivery type.

Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets machines with a specific description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopCondition

Gets machines with an outstanding desktop condition.

Valid values are:

- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupName

Gets machines from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets machines from a desktop group with a specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUUID

Gets machines from a desktop group with a specific UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Deprecated: Use AllocationType parameter.

Gets machines of a particular kind.

Valid values are Private, Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopUid

Gets the machine that corresponds to the desktop with the specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets machines with the specific DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DrainingUntilShutdown

Gets machines depending on whether they are draining until shutdown or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FaultState

Gets machines currently in the specified fault state.

Type:	MachineFaultState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FunctionalLevel

Gets machines with a specific FunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineId

Gets machines with the specific machine ID known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets machines with the specific machine name known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingServerName

Gets machines by the name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionName

Gets machines with the specific name of the hypervisor connection hosting them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets machines with the specific UID of the hypervisor connection hosting them.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypHypervisorConnectionUid

Gets machines with the specific UUID of the hypervisor connection hosting them.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets machines by configured icon. Note that machines with a null IconUid use the icon of the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets machines depending on whether their disk image is out of date or not (for machines provisioned using MCS only).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets machines by whether they are in maintenance mode or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets machines with a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsAssigned

Gets machines according to whether they are assigned or not. Machines may be assigned to one or more users or groups, a client IP address or a client endpoint name.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPhysical

Gets machines according to whether they can be power managed by XenDesktop or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsReserved

Gets machines that are reserved for special use, for example, for AppDisk preparation.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastAssignmentTime

Gets machines with the specific LastAssignmentTime.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionFailure

Gets machines with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the desktop yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

Type:	ConnectionFailureReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionTime

Gets machines on which a user session connection occurred at a specific time. This is the time at which the broker detected that the connection attempt either succeeded or failed.

Type:	DateTime
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionUser

Gets machines where a specific user name last attempted a connection (in the form 'domain\user').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastDeregistrationReason

Gets machines whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LastDeregistrationTime

Gets machines by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorReason

Gets machines with the specified last error reason.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastErrorTime

Gets machines with the specified last error time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets machines with a specific time that the hosting information was last updated or the completion of the last power action.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastPvdErrorReason

This property is no longer supported.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastPvdErrorTime

This property is no longer supported.

Type:	DateTime
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastRegistrationTime

Gets machines by the time that they last registered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoadIndex

Gets machines by their current load index.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MacAddress

Gets machines with a specific MAC address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineInternalState

Gets machines with the specified internal state.

Type:	MachineInternalState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUnavailableReason

Gets machines that corresponds to a particular MachineUnavailable Reason

- None - No detailed reason specified.
- LoadManagementInitializing - VDA load management logic currently initialising. Only occurs for multi-session capable VDAs.
- GctConnectionInitializing - VDA is still initializing control connection with NGS.
- AzureADJoinInitializing - VDA is still initializing Aad domain join.

Type:	String
Accepted values:	AzureADJoinInitializing, CtxUviDisabled, GctConnectionInitializing, IntuneEnrollPending, LoadManagementInitializing, None, PvdFailed, PvdInitializing

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaintenanceModeReason

Gets machines by the maintenance mode reason. Valid values are:

- None - Machine is not in maintenance mode.
- Administrator - Machine was manually placed in maintenance mode by an administrator.
- MaxFailedRegistrations - Machine was automatically placed in maintenance mode due to reaching the maximum failed registration limit.

Type:	MaintenanceModeReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NameLookupFailureCount

Tracks the number of consecutive directory lookup failures for this account.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets machines by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets machines by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PersistUserChanges

Gets machines by the location where the user changes are persisted.

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerActionPending

Gets machines depending on whether a power action is pending or not.

Valid values are $true or $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets machines with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningType

Gets machines that are in a catalog with a particular provisioning type. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical,

blade, VM,…).

- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedApplication

Gets machines with a specific application published to them (identified by its browser name).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PublishedName

Gets desktops with a specific published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PvdEstimatedCompletionTime

This property is no longer supported.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdPercentDone

This property is no longer supported.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdStage

This property is no longer supported.

Type:	PvdStage
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdUpdateStartTime

This property is no longer supported.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RegistrationState

Gets machines in a specific registration state.

Valid values are Unregistered, Initializing, Registered, and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScheduledReboot

Gets machines according to their current status with respect to any scheduled reboots (for either scheduled desktop group reboots or image rollout purposes). Valid values are:

- None - No reboot currently scheduled.
- Pending - Reboot scheduled but machine still available for use.
- Draining - Reboot scheduled. New logons are disabled, but reconnections to existing sessions are allowed.
- InProgress - Machine is actively being rebooted.
- Natural - Natural reboot in progress. Machine is awaiting a restart.

Type:	ScheduledReboot
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Gets machines configured with a particular SecureIcaRequired setting. Note that the machine setting of $null indicates that the desktop group value is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionAutonomouslyBrokered

Gets machines according to whether their current session is autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Session properties are always null for multi-session machines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionClientAddress

Gets machines with a specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionClientName

Gets machines with a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionClientVersion

Gets machines with a specific client version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionConnectedViaHostName

Gets machines with a specific incoming connection host name. This is usually a proxy or Citrix Access Gateway server.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-SessionConnectedViaIP

Gets machines with a specific incoming connection IP address.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionCount

Gets machines according to the total number of both pending and established user sessions on the machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionDeviceId

Gets machines with a specific client device ID.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionHardwareId

Gets machines with a specific client hardware ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionHidden

Gets machines depending on whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions using XenDesktop; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionKey

Gets machine running the session with a specified unique key.

Session properties are always null for multi-session machines.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionLaunchedViaHostName

Gets machines with a specific host name of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionLaunchedViaIP

Gets machines with a specific IP address of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-SessionProtocol

Gets machines with connections using a specific protocol, for example HDX, RDP, or Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionSecureIcaActive

Gets machines depending on whether the current session uses SecureICA or not.

Session properties are always null for multi-session machines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionsEstablished

Gets machines according to the number of established user sessions present on the machine.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSmartAccessTag

Gets machines where the session has the specific SmartAccess tag.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionsPending

Get machines according to the number of pending user sessions for the machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStartTime

Gets machines with a specific session start time.

Session properties are always null for multi-session machines.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Gets machines with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, NonBrokeredSession, and Unknown.

Session properties are always null for multi-session machines.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Gets machines whose sessions last changed state at a specific time.

Session properties are always null for multi-session machines.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-SessionSupport

Gets machines that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionType

Gets machines with a specific session state.

Session properties are always null for multi-session machines.

Type:	SessionType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets machines with a specific session UID ($null for no session).

Session properties are always null for multi-session machines.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserName

Gets machines with a specific user name for the current session (in the form 'domain\user').

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SessionUserSID

Gets machines with a specific SID of the current session user.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SID

Gets machines with a specific machine SID.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SummaryState

Gets machines with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, and InUse.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SupportedPowerAction

Gets machines that support the specified power action.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-Tag

Gets machines associated with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UUID

Gets machines with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMToolsState

Gets machines with a specific VM tools state.

Valid values are NotPresent, Unknown, NotStarted, and Running.

Type:	VMToolsState
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WillShutdownAfterUse

Gets machines depending on whether they shut down after use or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WillShutdownAfterUseReason

Gets machines by the will shutdown after use reason. Valid values are:

- None - Machine will not shutdown after use.
- ResetDiskImage - Machine will shutdown after use to reset its disk image.
- ScheduledNaturalReboot - Machine will shutdown after use as part of the scheduled natural reboot process.
- OnDemandNaturalReboot - Machine will shutdown after use as part of an on-demand natural reboot process.

Type:	WillShutdownAfterUseReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WindowsConnectionSetting

Gets machines according to their current Windows connection setting (logon mode). Valid values are:

- LogonEnabled - All logons are enabled.
- Draining - New logons are disabled, but reconnections to existing sessions are allowed.
- DrainingUntilRestart - Same as Draining, but setting reverts to LogonEnabled when machine next restarts.
- LogonDisabled - All logons and reconnections are disabled.

This is a Windows setting and is not controlled by XenDesktop. It applies only to multi-session machines; for single-session machines its value is always LogonEnabled.

Type:	WindowsConnectionSetting
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneHealthy

Gets machines located in the zone with the specified health state.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets machines located in the zone with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets machines located in the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String

Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Machine

Get-BrokerMachine returns an object for each matching desktop.

Notes

It is generally better to compare dates and times using -Filter and relative comparisons. See about_Broker_Filtering and the examples in this topic for more information.

Related Links

- about_Broker_Machines
- about_Broker_Filtering
- Group-BrokerMachine

Get-BrokerMachineCommand

March 11, 2024

Get the list of commands queued for delivery to a desktop.

Syntax

```
Get-BrokerMachineCommand
   [-Category <String>]
   [-CommandName <String>]
   [-CompletionTime <DateTime>]
   [-MachineName <String>]
   [-MachineUid <Int32>]
   [-Metadata <String>]
   [-RequestTime <DateTime>]
   [-SendDeadline <TimeSpan>]
   [-SendDeadlineTime <DateTime>]
   [-SendTrigger <MachineCommandTrigger>]
   [-SessionUid <Int64>]
   [-State <MachineCommandState>]
   [-User <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerMachineCommand
   -Uid <Int64>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Get the list of commands queued for delivery to a desktop. Commands are batched and can be configured to be delivered at various times during a desktop session's lifetime. Normally commands are sent within a few minutes of being queued, but it is also possible to queue a command for a user who is not currently logged on or a desktop that is currently switched off.

See about_Broker_Filtering for information about advanced filtering options.

——————————BrokerMachineCommand Object

The command object returned represents a command handled by a specific service on a desktop as determined by the Category property.

- Category (System.String)

  Category of the command.

- CommandData (System.Byte[])

  Additional binary data included when the command is sent.

- CommandName (System.String)

  Name of the command.

- CompletionTime (System.DateTime?)

  Time at which the command was sent, expired or canceled.

- DesktopGroupNames (System.String[])

  List of desktop group names that the command was restricted to.

- MachineName (System.String)

  Name of the machine this command is targeted at.

- MachineUid (System.Int32?)

  Unique ID of the machine this command is targeted at.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this command.

- RequestTime (System.DateTime)

  Time at which this command was created.

- SendDeadline (System.TimeSpan)

  Duration after which this command expires if it has not been sent yet.

- SendDeadlineTime (System.DateTime?)

  Time at which this command expires if it has not been sent yet.

- SendTrigger (Citrix.Broker.Admin.SDK.MachineCommandTrigger?)

  Event that triggers the sending of the command. Valid values are NextContact, Broker, LogOn, Logoff, Disconnect and Reconnect.

- SessionUid (System.Int64?)

  Unique ID of the session this command is targeted at.

- State (Citrix.Broker.Admin.SDK.MachineCommandState)

  Indicates whether the command is pending, sent, expired or canceled.

- Synchronous (System.Boolean)

  Flag that indicates if this is a synchronous command.

- Uid (System.Int64)

  Unique identifier of this machine command.

- User (System.String)

  Name of the user this command is targeted at.

Examples

EXAMPLE 1

Returns all pending, canceled, expired and sent commands.

```
Get-BrokerMachineCommand
```

EXAMPLE 2

Returns all queued commands.

```
Get-BrokerMachineCommand -State Pending
```

Parameters

-Uid

Get only the command with the specified unique identifier.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Category

Get only commands targeted to the specified service category.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CommandName

Get only commands with the specified command name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CompletionTime

Get only commands that entered the Sent, Failed, Canceled or Expired state at the specified time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Get only commands targeted to the specified machine.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineUid

Get only commands targeted to the specified machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RequestTime

Get only commands that were requested at the specified time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SendDeadline

Get only commands that expire after the specified time span.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SendDeadlineTime

Get only commands that have the specified deadline time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SendTrigger

Get only commands that are due to be sent when the specified trigger occurs. Valid values are NextContact, Broker, LogOn, Logoff, Disconnect and Reconnect.

Type:	MachineCommandTrigger
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Get only commands targeted to the specified session.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-State

Get only commands in the specified state. Valid values are Pending, Sent, Failed, Canceled and Expired.

Type:	MachineCommandState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-User

Get only commands targeted to the specified user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

No parameter is accepted from the input pipeline.

Outputs

Citrix.Broker.Admin.SDK.MachineCommand

Returns Command objects matching all specified selection criteria.

Related Links

- New-BrokerMachineCommand
- Remove-BrokerMachineCommand
- about_Broker_Filtering

Get-BrokerMachineConfiguration

March 11, 2024

Gets machine configurations defined for this site.

Syntax

```
Get-BrokerMachineConfiguration
   [[-Name] <String>]
   [-ConfigurationSlotUid <Int32>]
   [-LeafName <String>]
   [-Metadata <String>]
   [-ApplicationUid <Int32>]
   [-DesktopGroupUid <Int32>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerMachineConfiguration
   [-Uid] <Int32>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieves machine configurations matching the specified criteria. If no parameters are specified this cmdlet enumerates all machine configurations.

Machine configurations contain binary arrays of settings data that are managed using SDK snap-ins. Each machine configuration is associated with a configuration slot and referenced by Name. The configuration slot restricts the settings that can be held by the machine configuration. For example, only configurations for Citrix User Profile Manager can be associated with the "User Profile Manager"slot.

See about_Broker_Filtering for information about advanced filtering options.

——————————BrokerMachineConfiguration Object

The machine configuration object returned represents a named collection of related settings values that are applied to a desktop group.

- ApplicationUids (System.Int32[])

  List of application Uids that this machine configuration has been added to.

- ConfigurationSlotUid (System.Int32)

  Uid of the associated configuration slot.

- Description (System.String)

  Optional description of the machine configuration.

- DesktopGroupUids (System.Int32[])

  List of desktop group Uids that this machine configuration has been added to.

- LeafName (System.String)

  Name of this machine configuration.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Map of metadata associated with this machine configuration.

- Name (System.String)

  Unique "SlotName\MachineConfigurationName"for this machine configuration.

- Policy (System.Byte[])

  A binary array of encoded settings.

- Uid (System.Int32)

  Uid of this machine configuration.

Examples

EXAMPLE 1

Retrieves a list of every defined machine configuration.


```
Get-BrokerMachineConfiguration
```

EXAMPLE 2

Retrieves the machine configuration named “Receiver\Engineering”.

```
Get-BrokerMachineConfiguration -Name Receiver\Engineering
```

EXAMPLE 3

Retrieves a list of every machine configuration associated with the configuration slot named “UPM”
.

```
Get-BrokerMachineConfiguration -Name UPM\*
```

EXAMPLE 4

Retrieves a list of every machine configuration with a LeafName that starts with “Dept”, regardless of the associated configuration slot.

```
Get-BrokerMachineConfiguration -LeafName "Dept*"
```

Parameters

-Uid

Get only the machine configuration with the specified unique identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Get only the machine configuration with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConfigurationSlotUid

Get only the machine configurations associated with the specified configuration slot.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeafName

Get only the machine configurations that have the specified leaf name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Get only the machine configurations that have been assigned to the specified application.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Get only the machine configurations that have been assigned to the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.MachineConfiguration

Get-BrokerMachineConfiguration returns an object for each matching machine configuration.

Related Links

- New-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots
- about_Broker_Filtering

Get-BrokerMachineStartMenuShortcutIcon

March 11, 2024

Retrieves a Start Menu Shortcut icon from the specified machine.

Syntax

```
Get-BrokerMachineStartMenuShortcutIcon
   [-MachineName] <String>
   [-Path] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Retrieves the icon associated with a particular shortcut on a particular machine. This icon is usually used to help create a published application to access the shortcut.

Examples

EXAMPLE 1

This example retrieves all Start Menu Shortcuts from 'MyDomain\MyMachine', and then the icon for the first shortcut from the returned list. The icon is then associated with a published application called 'Notepad'.

```
$shortcuts =  Get-BrokerMachineStartMenuShortcuts -MachineName '
    MyDomain\MyMachine'
$encodedIconData =  Get-BrokerMachineStartMenuShortcutIcon -MachineName
     'MyDomain\MyMachine' -Path $shortcuts[0].ShortcutPath
$brokerIcon = New-BrokerIcon -EncodedIconData $encodedIconData
Set-BrokerApplication 'Notepad' -IconUid $brokerIcon.Uid
```

Parameters

-MachineName

Specify the name of the machine to use for icon retrieval for the specified shortcut path. The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-Path

The location of the shortcut in the specified machine whose icon is being fetched.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

Get-BrokerMachineStartMenuShortcutIcon generates a Base64 encoded string containing the icon for the specified shortcut. This can be used as input to New-BrokerIcon cmdlet.

Related Links

- about_Broker_Machines
- Get-BrokerMachine
- New-BrokerIcon

Get-BrokerMachineStartMenuShortcuts

March 11, 2024

Retrieves the Start Menu Shortcuts from the specified machine.

Syntax

```
Get-BrokerMachineStartMenuShortcuts
   [-MachineName] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieves the shortcuts defined for all the start menu items on a particular machine. The shortcuts obtained are from the 'All users'start menu; user-specific shortcuts are not found.

Examples

EXAMPLE 1

This example retrieves all Start Menu Shortcuts from 'MyDomain\MyMachine'.

```
$shortcuts = Get-BrokerMachineStartMenuShortcuts -MachineName 'MyDomain
   \MyMachine'
```

Parameters

-MachineName

Specify the name of the machine to use for shortcut retrieval. The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.StartMenuShorcut

Get-BrokerMachineStartMenuShortcuts generates an array of Citrix.Broker.Admin.SDK.StartMenuShortcut objects.

Related Links

- about_Broker_Machines

Get-BrokerMachineStatus

March 11, 2024

Gets detailed machine brokering and power management status

Syntax

```
Get-BrokerMachineStatus
    [[-MachineName] <String>]
    [-CatalogUid <Int32>]
    [-DesktopGroupUid <Int32>]
    [-IdlePoolStatus <String>]
    [-LaunchReadiness <String>]
    [-Sid <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerMachineStatus
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerMachineStatus cmdlet retrieves detailed internal status of machines relating to their readiness to accept new sessions or undertake power management operations. This information is for diagnostic purposes only.

————————BrokerMachineStatus Object

Each machine status object returned represents the detailed status of a physical or virtual machine configured in the site with respect to its readiness for brokering or power management.

- CatalogUid (System.Int32)

  The UID of the catalog containing the machine.

- CurrentSessions (System.Int32)

  The number of currently established sessions on the machine.

- DesktopGroupUid (System.Int32?)

  The UID of the desktop group containing the machine.

- DoNotShutdownBefore (System.DateTime)

  Time before which machine should not be shutdown by automatic power management.

- DrainingUntilReboot (System.Boolean)

  Whether no new sessions can be started on the machine as it is drained of sessions pending a reboot as part of a reboot cycle.

- DrainingUntilShutdown (System.Boolean)

  Whether no new sessions can be started on the machine as it is drained of sessions pending a shutdown.

- EffectiveLoadIndex (System.Int32?)

  The machine's effective load index (0 - 10000, RDS only).

- FreeSessions (System.Int32)

  The number of additional sessions the machine can accept before it reaches its configured session limit (always 0 or 1 for VDI machines).

- HasPendingPowerActions (System.Boolean)

  Whether the machine currently has outstanding, not yet started or completed, power actions.

- HasRegisteredInPeakTime (System.Boolean)

  Machine has registered in current peak time period.

- HasRegisteredSinceOutage (System.Boolean)

  Machine has registered since last recorded site outage.

- HypervisorInMaintenanceMode (System.Boolean)

  Whether the machine's hypervisor connection (if any) is in maintenance mode.

- IdlePoolStatus (System.String)

  Status of machine with respect to possible idle pool power management operations.

- IdleTimeBeforeUsageSecs (System.Int32?)

  Number of seconds for which the machine should ideally be left idle before it is selected for a new session (for example, after registration).

- ImageOutOfDate (System.Boolean)

  Whether the MCS master disk image currently being used by the machine is out of date.

- InMaintenanceMode (System.Boolean)

  Whether the machine is currently in maintenance mode.

- LastIdlePeriodStartTime (System.DateTime)

  Start time of the last idle period (see IdleTimeBeforeUsageSecs).

- LastPowerManagementActionUid (System.Int64?)

  UID of last power management action performed by this machine (persists after action has completed).

- LastPowerOnTime (System.DateTime)

  Time at which machine was last seen to power on.

- LastShutdownTime (System.DateTime)

  Time at which machine was last seen to power off.

- LastUserDrivenResetTime (System.DateTime)

  Time of last user 'restart'operation requested via CWA/SF.

- LaunchReadiness (System.String)

  Summary state of the machine's readiness to accept a new session.

- LogonsInProgress (System.Int32?)

  The number of session logons currently in progress on the machine (RDS only).

- MachineFaultState (System.String)

  Indicates any diagnostic fault state of the machine, or None otherwise.

- MachineName (System.String)

  The name of the machine (domaine
  ame).

- NeedsStart (System.Boolean)

  Whether the machine will be restarted when next convenient (a 'natural'reboot).

- NeedsStartReason (System.String)

  Reason why the machine will be restarted when next convenient (if any).

- PendingSessions (System.Int32)

  Number of outstanding sessions brokered to the machine that have not yet been established (client has not connected to the machine).

- PowerManagementActionUid (System.Int64?)

  UID of any power management action currently in progress for this machine.

- PowerOffExpectedSoonAfter (System.DateTime?)

  When not null and machine is On, time of last Shutdown/PowerOff action completed by machine and indicates that it should power off soon.

- PowerSinBinReleaseTime (System.DateTime?)

  If set, time after which automatic pool management may attempt to power on machine again following an earlier failure during power on.

- PowerState (System.String)

  The last reported power state of the machine.

- RebootCycleUid (System.Int64?)

  UID of any reboot cycle in which the machine is currently participating.

- SessionSupport (System.String)

  Whether the machine is a VDI (single session) or RDS (multi-session) machine.

- Sid (System.String)

  The Windows SID of the machine.

- SinBinReleaseTime (System.DateTime?)

  When set, indicates that the machine can not be selected for new sessions until the indicated time due to an earlier power management problem.

- TagUid (System.String[])

  List of tags associated with the machine (if any).

- TenantId (System.Guid?)

  Tenant ID associated with the machine (if any).

- Uid (System.Int32)

  The UID of the machine.

- Usage (System.String)

  The CVAD allocation type applicable to the machine.

- UserForcedLogOffGracePeriodExpiryTime (System.DateTime?)

  If set, time at which Autoscale will forcibly logoff remaining sessions on machine if users have not already logged off.

- UseVerticalScaling (System.Boolean?)

  If set, indicates that vertical load balancing is in use for machines in this desktop group.

- UseVerticalScalingForRdsLaunches (System.Boolean)

  Specifies the default for the site as to whether horizontal or vertical load balancing is in use for multi-session machines.

- WillShutdownAfterUse (System.Boolean)

  Whether the machine must be shutdown after current session ends.

- WillShutdownAfterUseReason (System.String)

  Reason why the machine must be shutdown after current session ends (if any).

- WindowsConnectionSetting (System.String)

  The Windows ConnectionSetting value reported on the machine (RDS only).

- WorkerState (System.String)

  Internal machine state.

- ZoneHealthy (System.Boolean)

  Whether the machine's zone is currently considered healthy.

- ZoneUid (System.Guid)

  The zone in which the machine resides.

Examples

Parameters

-Uid

Get status details of the machine with the specified UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets status details of machines having a name matching that specified.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogUid

Gets status details of machines in the specified catalog.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets status details of machines in the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdlePoolStatus

Gets status details of machines with the specified idle pool status.

Type:	String
Accepted values:	CanDrainAvailable, CanShutdownAvailable, CanShutdownUnregistered, CanShutdownUsed, CanStartup, Ignore
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchReadiness

Gets status detais of machines with the specified launch readiness.

Type:	String
Accepted values:	Draining, Later, Never, NotApplicable, Now, Soon
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Gets status details of the machine with the specified SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GetBrokerMachineStatus

Get-BrokerMachineStatus returns an object for each selected machine.

Related Links

Get-BrokerPowerTimeScheme

March 11, 2024

Gets power management time schemes for desktop groups.

Syntax

```
Get-BrokerPowerTimeScheme
   [[-Name] <String>]
   [-DesktopGroupUid <Int32>]
   [-Metadata <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerPowerTimeScheme
   [-Uid] <Int32>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Finds power time schemes matching the specified search criteria. Each desktop group in the site can have a number of power time schemes associated with it, and these time schemes control how the power states of machines in the group are managed.

If no search criteria are specified all power time schemes for all desktop groups are obtained.

Each power time scheme covers one or more days of the week, and defines which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For any day of the week not covered by any power time scheme, it is assumed that all hours are off-peak and no pool size management is required for any of the hours.

PeakHours is deprecated, and PeakHalfHours should be used instead.

PoolSize is deprecated, and PoolSizeHalfHours should be used instead.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

——————————BrokerPowerTimeScheme Object

The BrokerPowerTimeScheme object represents a power time scheme, defining peak/off-peak hours and idle pool sizes for desktop groups. It contains the following properties:

- DaysOfWeek (Citrix.Broker.Admin.SDK.TimeSchemeDays)

  The days of the week for which this scheme applies to (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Weekdays, Weekend).

- DesktopGroupUid (System.Int32)

  The desktop group that this time scheme is for.

- DisplayName (System.String)

  The name of this time scheme, as displayed in the Studio console.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this power time scheme.

- Name (System.String)

  The unique name of this time scheme.

- PeakHalfHours (System.Boolean[])

A set of 48 boolean flag values, one for each hour half of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. If the flag is $true it means that the associated half hour of the day is considered a peak time; if $false it means that it is considered off-peak.

- PeakHours (System.Boolean[])

  A set of 24 boolean flag values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag is $true it means that the associated hour of the day is considered a peak time; if it is $false it means that it is considered off-peak.

- PoolSize (System.Int32[])

  For single-session desktop groups, a set of 24 integer values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. For multi-session desktop groups, a set of 48 integer values, one for each half hour of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such hours.

- PoolSizeHalfHours (System.Int32[])

  A set of 48 integer values, one for each half hour of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such half hours.

- PoolUsingPercentage (System.Boolean?)

  A boolean flag to indicate whether the integer values in the pool size array are to be treated as absolute values (if this value is $false) or as percentages of the number of machines in the desktop group (if this value is $true).

- Uid (System.Int32)

  Unique internal identifier of a time scheme.

Examples

EXAMPLE 1

Fetches all known power time schemes for all desktop groups in the site.

```
Get-BrokerPowerTimeScheme
```

EXAMPLE 2

Fetches all the power time schemes for the desktop group called 'Sales Desktops'.

```
Get-BrokerPowerTimeScheme -DesktopGroupUid ( Get-BrokerDesktopGroup '
    Sales Desktops' ).Uid
```

Parameters

-Uid

Gets only the power time scheme with the specified Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets only power time schemes with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets only the power time schemes associated with the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.PowerTimeScheme

Get-BrokerPowerTimeScheme returns all power time schemes that match the specified selection criteria.

Related Links

- about_Broker_PowerManagement
- about_Broker_Filtering
- New-BrokerPowerTimeScheme
- Set-BrokerPowerTimeScheme
- Remove-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme

Get-BrokerPrivateDesktop

March 11, 2024

Get private desktops configured for this site.

Syntax

```
Get-BrokerPrivateDesktop
    [[-MachineName] <String>]
    [-AgentVersion <String>]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-ColorDepth <ColorDepth>]
    [-ControllerDNSName <String>]
    [-Description <String>]
    [-DesktopGroupUid <Int32>]
    [-DNSName <String>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAssigned <Boolean>]
    [-LastDeregistrationReason <DeregistrationReason>]
```



```
    [-LastDeregistrationTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-PowerState <PowerState>]
    [-PublishedName <String>]
    [-RegistrationState <RegistrationState>]
    [-SecureIcaRequired <Boolean>]
    [-SID <String>]
    [-Tag <String>]
    [-WillShutdownAfterUse <Boolean>]
    [-AssignedUserSID <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerPrivateDesktop
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet is deprecated, please use the Get-BrokerMachine cmdlet instead.

Retrieve private desktops matching the specified criteria. If no parameters are specified, this cmdlet enumerates all private desktops.

Get-BrokerPrivateDesktop returns configuration information only for private desktops (a DesktopKind of 'Private'). For more state information about desktops, or other types of desktop, use the Get-BrokerMachine cmdlet instead.

For information about advanced filtering options, see about_Broker_Filtering; for more information about desktops, see about_Broker_Desktops.

——————————BrokerPrivateDesktop Object

Private desktops are machines that have been configured with a DesktopKind of 'Private'. They are allocated to either a user/users or a client name/address (but cannot be allocated to both).

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

- AssignedClientName (System.String)

  Client name the desktop has been assigned to.

- AssignedIPAddress (System.String)

  IP Address the desktop has been assigned to.

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth?)

  The color depth setting configured on the desktop, possible values are:
  $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

- ControllerDNSName (System.String)

  The DNS host name of the controller that the desktop is registered to.

- Description (System.String)

  Description of the private desktop.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group the desktop has been assigned to.

- DNSName (System.String)

  The DNS host name of the desktop.

- HostedMachineId (System.String)

  Unique ID within the hosting unit of the target managed desktop.

- HostedMachineName (System.String)

  The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

- HostingServerName (System.String)

  DNS name of the hypervisor that is hosting the desktop if managed.

- HypervisorConnectionUid (System.Int32?)

  The UID of the hypervisor connection that the desktop has been assigned to, if managed.

- IconUid (System.Int32?)

  The UID of the desktop's icon that is displayed in StoreFront. If this is $null then the desktop will use the icon specified by the desktop group.

- InMaintenanceMode (System.Boolean)

  Denotes whether the desktop is in maintentance mode.

- IPAddress (System.String)

  The IP address of the desktop.

- IsAssigned (System.Boolean)

  Denotes whether a private desktop has been assigned to a user/users, or a client name/address. Users can be assigned explictitly or by assigning on first use of the desktop.

- LastDeregistrationReason (Citrix.Broker.Admin.SDK.DeregistrationReason?)

  The reason for the last deregistration of the desktop with the broker. Possible values are: AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- LastDeregistrationTime (System.DateTime?)

  Time of the last deregistration of the desktop from the controller.

- LastHostingUpdateTime (System.DateTime?)

  Time of last update to any hosting data for this desktop reported by the hypervisor connection.

- MachineName (System.String)

  DNS host name of the machine associated with the desktop.

- OSType (System.String)

  A string that can be used to identify the operating system that is running on the desktop.

- OSVersion (System.String)

  A string that can be used to identify the version of the operating system running on the desktop, if known.

- PowerState (Citrix.Broker.Admin.SDK.PowerState)

  The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, Resuming.

- PublishedName (System.String)

  The name of the desktop that is displayed in StoreFront, if the desktop is published.

- RegistrationState (Citrix.Broker.Admin.SDK.RegistrationState)

Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

- SecureIcaRequired (System.Boolean?)

  Flag indicating whether SecureICA is required or not when starting a session on the desktop.

- SID (System.String)

  Security identifier of the private desktop.

- Uid (System.Int32)

  Unique identifier of the private desktop.

- WillShutdownAfterUse (System.Boolean)

  Flag indicating whether this desktop is tainted and will be shutdown after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.
  Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Examples

EXAMPLE 1

Get all private desktops that are turned on, or are turning on (assuming unmanaged desktops are powered on).

```
$list = 'Unmanaged','On','TurningOn','Resuming'
Get-BrokerPrivateDesktop -Filter {
  PowerState -in $list  }
  | ft -a DNSName,PowerState
```

EXAMPLE 2

Retrieve all private desktops tagged with the 'TestTag'tag.

```
Get-BrokerPrivateDesktop -Tag TestTag
```

Parameters

-Uid

Gets desktops by Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets desktops by machine name (in the form 'domain\machine').

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AgentVersion

Gets desktops with a specific Citrix Virtual Delivery Agent (VDA) version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedClientName

Gets desktops assigned to a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssignedIPAddress

Gets desktops assigned to a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ColorDepth

Gets desktops configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets desktops by the DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets desktops by description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets desktops from a desktop group with a specific Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets desktops by DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineId

Gets desktops by the machine id known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets desktops by the machine name known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingServerName

Gets desktops by the name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets desktops by the uid of the hosting hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets desktops by configured icon. Note that desktops with a $null IconUid use the icon of the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets desktops by the InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Get desktops by their IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsAssigned

Gets desktops depending on whether they are assigned or not. Private desktops can be assigned to either a user/users or client names/addresses.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationReason

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationTime

Gets desktops by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets desktops by the time that the hosting information was last updated.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets desktops by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets desktops by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PowerState

Gets desktops by power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Gets desktops by published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RegistrationState

Gets desktops by registration state.

Valid values are Registered, Unregistered, and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Gets desktops configured with a particular SecureIcaRequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Gets desktops by machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Tag

Gets desktops tagged with the given tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-WillShutdownAfterUse

Gets desktops depending on whether they will be automatically shut down when the current session ends or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedUserSID

Gets desktops with the given assigned user (specified by SID).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.PrivateDesktop

Get-BrokerPrivateDesktop returns an object for each matching private desktop.

Notes

To compare dates/times, use -Filter and relative comparisons. See about_Broker_Filtering for details.

Related Links

- about_Broker_Filtering
- about_Broker_Desktops
- Set-BrokerPrivateDesktop

Get-BrokerProjectedAutoscaleMachines

March 11, 2024

Gets the projected number of machines that Autoscale will keep powered on over the specified period

Syntax

```
Get-BrokerProjectedAutoscaleMachines
   [-DesktopGroupUid] <Int32>
   [-FromDate <String>]
   [-NumberOfDays <Int32>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Get-BrokerProjectedAutoscaleMachines cmdlet retrieves the projected number of machines that Autoscale will keep powered on for each half hour over a specified period.

————————BrokerProjectedAutoscaleMachines Object

The projected Autoscale machines object returned represents the detailed projection of the number of machines that Autoscale will keep powered on for each half hour over a specified period.

- DesktopGroupUid (System.Int32)

  The UID of the desktop group

- ManagedMachineCount (System.Int32)

  The number of machines in the desktop group that would be managed by Autoscale.

- ProjectedMachines (System.String)

  The number of machines that Autoscale will keep powered on for each half hour over the specified period.

Examples

Parameters

-DesktopGroupUid

Gets projected Autoscale machines in the specified desktop group.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FromDate

Gets projected Autoscale machines for the period starting at the specified date (YYYY-MM-DD).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-NumberOfDays

Gets projected Autoscale machines for the period consisting of the specified number of days.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ProjectedAutoscaleMachines

Get-BrokerProjectedAutoscaleMachines returns a ProjectedAutoscaleMachines object.

Related Links

Get-BrokerRebootCycle

March 11, 2024

Gets one or more reboot cycles.

Syntax

```
Get-BrokerRebootCycle
    [-CatalogName <String>]
    [-CatalogUid <Int32>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-EndTime <DateTime>]
    [-IgnoreMaintenanceMode <Boolean>]
    [-MachinesCompleted <Int32>]
    [-MachinesFailed <Int32>]
    [-MachinesInProgress <Int32>]
    [-MachinesPending <Int32>]
    [-MachinesSkipped <Int32>]
    [-Metadata <String>]
    [-RebootDuration <Int32>]
    [-RebootScheduleName <String>]
    [-RebootScheduleUid <Int32>]
    [-RestrictToTag <String>]
    [-StartTime <DateTime>]
    [-State <RebootCycleState>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerRebootCycle
   -Uid <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

The Get-BrokerRebootCycle cmdlet is used to enumerate reboot cycles that match all of the supplied criteria.

See about_Broker_Filtering for information about advanced filtering options.

————————BrokerRebootCycle Object

The reboot cycle object returned represents a single occurrence of the process of rebooting a portion (or all) of the machines in a desktop group.

- CatalogName (System.String)

  Name of the catalog whose machines are rebooted by this cycle if the cycle is associated with a catalog.

- CatalogUid (System.Int32?)

  Uid of the catalog whose machines are rebooted by this cycle if the cycle is associated with a catalog.

- DesktopGroupName (System.String)

  Name of the desktop group whose machines are rebooted by this cycle.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group whose machines are rebooted by this cycle.

- EndTime (System.DateTime?)

  Time at which this cycle was completed, canceled or abandoned.

- IgnoreMaintenanceMode (System.Boolean)

  Boolean value to optionally reboot machines in maintenance mode

- MachinesCompleted (System.Int32)

  Number of machines successfully rebooted by this cycle.

- MachinesFailed (System.Int32)

  Number of machines issued with reboot requests where either the request failed or the operation did not complete within the allowed time.

- MachinesInProgress (System.Int32)

  Number of machines issued with reboot requests but which have not yet completed the operation.

- MachinesPending (System.Int32)

  Number of outstanding machines to be rebooted during the cycle but on which processing has not yet started.

- MachinesSkipped (System.Int32)

  Number of machines scheduled for reboot during the cycle but which were not processed either because the cycle was canceled or abandoned or because the machine was unavailable for reboot processing throughout the cycle.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Map of metadata associated with this cycle.

- RebootDuration (System.Int32)

  Approximate maximum number of minutes over which the reboot cycle runs.

- RebootScheduleName (System.String)

  Name of the Reboot Schedule which triggered this cycle if the cycle is associated with a reboot schedule.

- RebootScheduleUid (System.Int32?)

  Uid of the Reboot Schedule which triggered this cycle if the cycle is associated with a reboot schedule.

- RestrictToTag (System.String)

  An optional Tag which limits the reboot cycle to machines within the desktop group with the specified tag.

- StartTime (System.DateTime)

  Time of day at which this reboot cycle was started.

- State (Citrix.Broker.Admin.SDK.RebootCycleState)

  The execution state of this cycle.

- Uid (System.Int64)

  Unique ID of this reboot cycle.

- WarningDuration (System.Int32)

  Number of minutes to display the warning message for.

- WarningMessage (System.String)

  Warning message to display to users in active sessions prior to rebooting the machine.

- WarningRepeatInterval (System.Int32)

  Number of minutes to wait before showing the reboot warning message again.

- WarningTitle (System.String)

  Title of the warning message dialog.

Examples

EXAMPLE 1

Enumerate all reboot cycles.

```
Get-BrokerRebootCycle
```

EXAMPLE 2

Enumerates all reboot cycles that have successfully completed.

```
Get-BrokerRebootCycle -State Completed
```

EXAMPLE 3

Enumerates all reboot cycles related to the desktop group named CallCenter.

```
Get-BrokerRebootCycle -DesktopGroupName CallCenter
```

Parameters

-Uid

Gets reboot cycles that have the specified Uid.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets reboot cycles that relate to the named catalog.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogUid

Gets reboot cycles that relate to the catalog with a particular Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets reboot cycles that relate to the named desktop group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets reboot cycles that relate to the desktop group with a particular Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndTime

Gets reboot cycles that have the specified time at which the reboot cycle was completed, canceled or abandoned.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreMaintenanceMode

Boolean value to optionally reboot machines in maintenance mode

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesCompleted

Gets reboot cycles that have the specified count of machines successfully rebooted during the cycle.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesFailed

Gets reboot cycles that have the specified count of machines issued with reboot requests where either the request failed or the operation did not complete within the allowed time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesInProgress

Gets reboot cycles that have the specified count of machines issued with reboot requests but which have not yet completed the operation.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MachinesPending

Gets reboot cycles that have the specified count of outstanding machines to be rebooted during the cycle but on which processing has not yet started.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesSkipped

Gets reboot cycles that have the specified count of machines scheduled for reboot during the cycle but which were not processed either because the cycle was canceled or abandoned or because the machine was unavailable for reboot processing throughout the cycle.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Gets reboot cycles that have the specified approximate maximum duration in minutes over which the reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootScheduleName

Gets reboot cycles which were triggered by the named reboot schedule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-RebootScheduleUid

Gets reboot cycles which were triggered by the reboot schedule with aparticular Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

An optional Tag which limits the reboot cycle to machines within the desktop group with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-StartTime

Gets reboot cycles that have the specified time at which the reboot cycle started.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-State

Gets reboot cycles that have the specified overall state of the reboot cycle. Valid values are Initializing, Active, Completed, Canceled, and Abandoned.

Type:	RebootCycleState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RebootCycle

Returns matching reboot cycles.

Related Links

- Start-BrokerRebootCycle
- Start-BrokerDesktopGroupRebootCycle
- Stop-BrokerRebootCycle
- about_Broker_Filtering

Get-BrokerRebootSchedule

March 11, 2024

Gets one or more reboot schedules.

Syntax

```
Get-BrokerRebootSchedule
    [[-DesktopGroupName] <String>]
    [-Active <Boolean>]
    [-Day <RebootScheduleDays>]
    [-Enabled <Boolean>]
    [-Frequency <RebootScheduleFrequency>]
    [-RebootDuration <Int32>]
    [-StartTime <TimeSpan>]
    [-WarningRepeatInterval <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerRebootSchedule
    [-DesktopGroupUid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerRebootSchedule cmdlet is used to enumerate desktop group reboot schedules that match all of the supplied criteria.

A reboot schedule can be configured to cause all of the machines in a desktop group to be rebooted at a particular time each day or each week, with the reboot of the individual machines spread out over the duration of the whole reboot cycle. A specific warning message can be configured to be displayed to users who are running sessions on the machines being rebooted. Note that each desktop group can only have a single reboot schedule configured.

See about_Broker_Filtering for information about advanced filtering options.

——————BrokerRebootSchedule Object

The reboot schedule object returned represents a regularly scheduled reboot of machines in a desktop group.

- Active (System.Boolean)

  True if there is an active reboot cycle for this schedule, false otherwise.

- Day (Citrix.Broker.Admin.SDK.RebootScheduleDays?)

  When the frequency is weekly, day of the week on which the schedule reboot starts.

- DesktopGroupName (System.String)

  Name of the desktop group rebooted by this schedule.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group rebooted by this schedule.

- Enabled (System.Boolean)

  True if this schedule is currently enabled, false otherwise.

- Frequency (Citrix.Broker.Admin.SDK.RebootScheduleFrequency)

  Whether the schedule runs daily or weekly.

- RebootDuration (System.Int32)

  Approximate maximum number of minutes over which the scheduled reboot cycle runs.

- StartTime (System.TimeSpan)

  Time of day at which the scheduled reboot cycle starts.

- WarningDuration (System.Int32)

  Number of minutes to display the warning message for.

- WarningMessage (System.String)

  Warning message to display to users in active sessions prior to rebooting the machine.

- WarningRepeatInterval (System.Int32)

  Number of minutes to wait before displaying the warning message again.

- WarningTitle (System.String)

  Title of the warning message dialog.

Examples

EXAMPLE 1

Enumerates all of the reboot schedules.

```
1  Get-BrokerRebootSchedule
```


EXAMPLE 2

Enumerates all disabled reboot schedules that are scheduled to run daily.

```
Get-BrokerRebootSchedule -Enabled $false -Frequency Daily
```

EXAMPLE 3

Returns the unique reboot schedule for the desktop group having the Uid 11.

```
Get-BrokerRebootSchedule -DesktopGroupUid 11
```

Parameters

-DesktopGroupUid

Gets the reboot schedule for the desktop group having this Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets the reboot schedule for the desktop group having this name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Active

Gets desktop group reboot schedules according to whether they are currently active or not. A schedule is active if there is a reboot cycle currently running that was started as a result of the schedule.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Day

Gets the reboot schedules set to run on the specified day (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets the reboot schedules with the specified Enabled value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Frequency

Gets the reboot schedules with the specified frequency (either Weekly or Daily).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Gets the reboot schedules with the specified duration.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Gets the reboot schedules with the specified start time (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Gets the reboot schedules with the specified warning repeat interval.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32

Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RebootSchedule

Returns matching reboot schedules.

Related Links

- Set-BrokerRebootSchedule
- New-BrokerRebootSchedule
- Remove-BrokerRebootSchedule
- Get-BrokerRebootCycle
- about_Broker_Filtering

Get-BrokerRebootScheduleV2

March 11, 2024

Gets one or more reboot schedules.

Syntax

```
Get-BrokerRebootScheduleV2
    [[-Name] <String>]
    [-Active <Boolean>]
    [-Day <RebootScheduleDays>]
    [-DayInMonth <RebootScheduleDays>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-Enabled <Boolean>]
    [-Frequency <RebootScheduleFrequency>]
    [-FrequencyFactor <Int32>]
    [-IgnoreMaintenanceMode <Boolean>]
    [-MaxOvertimeStartMins <Int32>]
    [-MetadataKey <String>]
    [-Metadata <String>]
    [-RebootDuration <Int32>]
    [-RestrictToTag <String>]
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-UseNaturalReboot <Boolean>]
    [-WarningRepeatInterval <Int32>]
    [-WeekInMonth <RebootScheduleWeeks>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerRebootScheduleV2
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerRebootScheduleV2 cmdlet is used to enumerate desktop group reboot schedules that match all of the supplied criteria.

A reboot schedule can be configured to cause all of the machines in a desktop group to be rebooted at a particular time each day or each week, with the reboot of the individual machines spread out over the duration of the whole reboot cycle. A specific warning message can be configured to be displayed to users who are running sessions on the machines being rebooted.

See about_Broker_Filtering for information about advanced filtering options.

——————————BrokerRebootScheduleV2 Object

The reboot schedule object returned represents a regularly scheduled reboot of machines in a desktop group.

- Active (System.Boolean)

  True if there is an active reboot cycle for this schedule, false otherwise.

- Day (Citrix.Broker.Admin.SDK.RebootScheduleDays?)

  When the frequency is weekly, days of the week on which the schedule reboot starts.

- DayInMonth (Citrix.Broker.Admin.SDK.RebootScheduleDays?)

  When the frequency is monthly, day in the month on which the schedule reboot starts.

- Description (System.String)

  An optional description for the reboot schedule.

- DesktopGroupName (System.String)

  Name of the desktop group rebooted by this schedule.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group rebooted by this schedule.

- Enabled (System.Boolean)

  True if this schedule is currently enabled, false otherwise.

- Frequency (Citrix.Broker.Admin.SDK.RebootScheduleFrequency)

  Whether the schedule runs daily or weekly or monthly.

- FrequencyFactor (System.Int32)

  The frequency factor for the reboot schedule. It indicates how often the schedule should run with respect to the frequency. For example, if the FrequencyFactor is set to 2, it means every two days from the StartDate when the Frequency is Daily, every two weeks from the StartDate when the Frequency is Weekly, and similarly for monthly.

- IgnoreMaintenanceMode (System.Boolean)

  Boolean value to reboot machines in maintenance mode

- MaxOvertimeStartMins (System.Int32)

  Maximum delay in minutes after which the scheduled reboot will not take place

- MetadataKeys (System.String[])

  All key names of metadata items associated with this application.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Metadata for this application.

- Name (System.String)

  Name of the reboot schedule.

- RebootDuration (System.Int32)

  Approximate maximum number of minutes over which the scheduled reboot cycle runs.

- RestrictToTag (System.String)

  If set the reboot schedule only applies to machines in the desktop group with the specified tag.

- StartDate (System.String)

  The date on which the first schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD).

- StartTime (System.TimeSpan)

  Time of day at which the scheduled reboot cycle starts.

- Uid (System.Int32)

  Uid of the reboot schedule.

- UseNaturalReboot (System.Boolean)

  Boolean value indicating whether the machines should reboot whenever they happen to have no sessions, rather than at equally spaced times within the cycle duration.

- WarningDuration (System.Int32)

  Number of minutes to display the warning message for.

- WarningMessage (System.String)

  Warning message to display to users in active sessions prior to rebooting the machine.

- WarningRepeatInterval (System.Int32)

  Number of minutes to wait before displaying the warning message again.

- WarningTitle (System.String)

  Title of the warning message dialog.

- WeekInMonth (Citrix.Broker.Admin.SDK.RebootScheduleWeeks?)

  When the frequency is monthly, week in the month on which the schedule reboot starts.

Examples

EXAMPLE 1

Enumerates all of the reboot schedules.

```
Get-BrokerRebootScheduleV2
```

EXAMPLE 2

Enumerates all disabled reboot schedules that are scheduled to run daily.

```
Get-BrokerRebootScheduleV2 -Enabled $false -Frequency Daily
```

EXAMPLE 3

Returns the reboot schedules for the desktop group having the Uid 11.

```
Get-BrokerRebootScheduleV2 -DesktopGroupUid 11
```

Parameters

-Uid

Gets the reboot schedule with the specified Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets the reboot schedule with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Active

Gets desktop group reboot schedules according to whether they are currently active or not. A schedule is active if there is a reboot cycle currently running that was started as a result of the schedule.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Day

Gets the reboot schedules set to run on the specific days of week (one or more of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DayInMonth

Gets the reboot schedules set to run on the specified day in month.

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets the reboot schedules for the desktop group having this name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets the reboot schedules for the desktop group having this Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Gets the reboot schedules with the specified Enabled value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Frequency

Gets the reboot schedules with the specified frequency (either Weekly or Daily or Monthly).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FrequencyFactor

Gets the reboot schedules with the specified frequency factor.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreMaintenanceMode

Boolean value to reboot machines in maintenance mode

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MetadataKey

All key names of metadata items associated with this application.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Gets the reboot schedules with the specified duration.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

Gets the reboot schedules with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-StartDate

Gets the reboot schedules with the specified start date (YYY-MM-DD).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-StartTime

Gets the reboot schedules with the specified start time (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseNaturalReboot

Gets the reboot schedules with the specified UseNaturalReboot value.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Gets the reboot schedules with the specified warning repeat interval.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WeekInMonth

Gets the reboot schedules with the specified week in a month.

Type:	RebootScheduleWeeks
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RebootScheduleV2

Returns matching reboot schedules.

Related Links

- Set-BrokerRebootScheduleV2
- New-BrokerRebootScheduleV2
- Remove-BrokerRebootScheduleV2
- Rename-BrokerRebootScheduleV2
- Get-BrokerRebootCycle
- about_Broker_Filtering

Get-BrokerRemotePCAccount

March 11, 2024

Get RemotePCAccount entries configured for this site.

Syntax

```
Get-BrokerRemotePCAccount
   [-AllowSubfolderMatches <Boolean>]
   [-CatalogUid <Int32>]
   [-OU <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerRemotePCAccount
   -Uid <Int32>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieves RemotePCAccounts matching the specified criteria. If no parameters are specified this cmdlet enumerates all RemotePCAccounts. Each RemotePCAccount object defines a set of machines

either by machine name patterns or by where the machines are placed in Active Directory, and which RemotePC catalog the machines are to be associated with when they are discovered.

————————BrokerRemotePCAccount Object

RemotePCAccounts define a set of machines either by machine name patterns or by where the machines are placed in Active Directory, and which RemotePC catalog the machines are to be associated with when they are discovered.

- AllowSubfolderMatches (System.Boolean)

  Specifies whether machines subfolders of specified AD OUs are to be considered part of the RemotePCAccount.

- CatalogUid (System.Int32)

  The Uid of the RemotePC catalog to which machines in the RemotePCAccount automatically join during registration.

- MachinesExcluded (System.String[])

  A list of machines which are to be excluded from the RemotePCAccount. Wildcard matching is supported.

- MachinesIncluded (System.String[])

  A list of machines which are to be included in the RemotePCAccount. Wildcard matching is supported.

- OU (System.String)

  Machines within this specified AD OU are considered part of the RemotePCAccount, unless they are in they match the MachinesExcluded

- Uid (System.Int32)

  The Uid of the RemotePCAccount object.

Examples

EXAMPLE 1

Find all RemotePCAccounts.

```
1 Get-BrokerRemotePCAccount
```

EXAMPLE 2

Find RemotePCAccounts belonging to Remote PC catalog 42.


```
1  Get-BrokerRemotePCAccount -CatalogUid 42
```

Parameters

-Uid

Gets the RemotePCAccount with the specified unique ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowSubfolderMatches

Gets RemotePCAccounts with the specified value of AllowSubfolderMatches.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUid

Gets RemotePCAccounts belonging to the specified Remote PC catalog.

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OU

Gets the RemotePCAccount with the specified OU.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RemotePCAccount

Get-BrokerRemotePCAccount returns an object for each matching RemotePCAccount.

Related Links

- about_Broker_RemotePC
- New-BrokerRemotePCAccount
- Set-BrokerRemotePCAccount
- Remove-BrokerRemotePCAccount

Get-BrokerResource

March 11, 2024

Gets resources that a user can broker connections to.

Syntax

```
Get-BrokerResource
    [-User] <String>
    [-Groups <String[]>]
    [-TenantId <Guid>]
    [-ClientName <String>]
    [-ClientIP <String>]
    [-ViaAG <Boolean>]
    [-SmartAccessTags <String[]>]
    [-AppProtectionCapable <Boolean>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve a list of resources that a user has access to, taking into account the site access policy, configuration of desktop groups, assignments, entitlements, and applications.

What a user has access to depends on a number of attributes:

- User's name or security identifier.
- Groups that the user is a member of (names or security identifiers).
- IP address of the client the user connects from.
- Name of the client that the user connects from.
- Whether the user is connecting via Citrix Access Gateway.
- SmartAccess tags when connecting via Citrix Access Gateway.

You must always specify the user's name or security identifier, but you will not always be able to predict what some of the other values will be. By omitting these values the corresponding access checks are ignored.

Consider for example, a site configuration that uses IP address ranges to allow access to private desktop A when connecting from the local network and private desktop B when connecting from home. Running this cmdlet without specifying a client IP address would return both A and B.

The output of this cmdlet depends on the available resources:

- Assigned private desktops are returned as PrivateDesktop objects.
- Shared desktops are returned as EntitlementPolicyRule objects.
- Assign-On-First-Use desktops that have not been assigned yet are returned as AssignmentPolicyRule objects.
- Application resources produce Application objects.

If more than one type of resource is available, the output pipeline contains a mixture of the above objects, in no particular order.

Only resources accessible based on the specified parameters, and visible to the administrator running this cmdlet are returned.

Examples

EXAMPLE 1

List resources visible by User1 assuming membership of a couple of groups.

```
Get-BrokerResource -User MYDOMAIN\User1 -Groups MYDOMAIN\Accounts,
    MYDOMAIN\Managers
```

EXAMPLE 2

Get all of the desktop groups supporting the resources accessible by User1, outputting the uid and name of each desktop group.

```
[int[]]$groups = (Get-BrokerResource -User MYDOMAIN\User1 | %{
  $_.DesktopGroupUid  }
 )
Get-BrokerDesktopGroup -Filter {
  Uid -in $groups  }
  -Property Uid,Name
```

Parameters

-User

Gets resources given the specified user name or security identifier.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-Groups

Get resources accessible given a list of group names or security identifiers.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with the user and groups. Can only be used in multitenant sites.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientName

Get resources given the specified client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientIP

Get resources given the specified client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ViaAG

Gets resources given the specified ViaAG setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SmartAccessTags

Get resources given the specified SmartAccess tags.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionCapable

Get Resources given the specified app protection capability of the client.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Application

Get-BrokerResource returns an Application object for each accessible application.

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

Get-BrokerResource returns an AssignmentPolicyRule object for each accessible Assign-On-First-Use desktop.

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

Get-BrokerResource returns an EntitlementPolicyRule object for each accessible entitlement to a shared desktop.

Citrix.Broker.Admin.SDK.PrivateDesktop

Get-BrokerResource returns a PrivateDesktop for each accessible assigned private desktop.

Related Links

- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_Desktops
- about_Broker_Applications

Get-BrokerScopedObject

March 11, 2024

Gets the details of the scoped objects for the Broker Service.

Syntax

```
Get-BrokerScopedObject
   [-Description <String>]
   [-ObjectId <String>]
   [-ObjectName <String>]
   [-ObjectType <ScopedObjectType>]
   [-ScopeName <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Get-BrokerScopedObject
   -ScopeId <Guid>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

——————————BrokerScopedObject Object

A scoped, or scopeable object configured in the Broker.

- Description (System.String)

  Description of the object (possibly $null if the object type does not have a description).

- ObjectId (System.String)

  Unique identifier of the object.

- ObjectName (System.String)

  Display name of the object.

- ObjectType (Citrix.Broker.Admin.SDK.ScopedObjectType)

  Type of the object this entry relates to.

- ScopeId (System.Guid?)

  Specifies the unique identifier of the scope.

- ScopeName (System.String)

  Specifies the display name of the scope.

Examples

EXAMPLE 1

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in

any scope. The ScopeId and ScopeName values returned are null in the final record.

```
Get-BrokerScopedObject -ObjectType Scheme

    ScopeId     : eff6f464-f1ee-4442-add3-99982e0cec01
    ScopeName   : Sales
    ObjectType  : Scheme
    ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
    ObjectName  : MyExampleScheme
    Description : Test scheme

    ScopeId     : 304e0fa7-d390-47f0-a94f-7e956a324c41
    ScopeName   : Finance
    ObjectType  : Scheme
    ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
    ObjectName  : MyExampleScheme
    Description : Test scheme

    ScopeId     :
    ScopeName   :
    ObjectType  : Scheme
    ObjectId    : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
    ObjectName  : AnotherScheme
    Description : Another scheme in no scopes
```

Parameters

-ScopeId

Gets scoped object entries for the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets scoped object entries for objects with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ObjectId

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ObjectName

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ObjectType

Gets scoped object entries for objects of the given type.

Type:	ScopedObjectType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets scoped object entries with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Sdk.ScopedObject

The Get-BrokerScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Specifies the unique identifier of the scope.

ScopeName <String>

Specifies the display name of the scope.

ObjectType <ScopedObjectType>

Type of the object this entry relates to.

ObjectId <String>

Unique identifier of the object.

ObjectName <String>

Display name of the object

Description <String>

Description of the object (possibly $null if the object type does not have a description).

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts

Get-BrokerServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Broker Service on the controller.

Syntax

```
Get-BrokerServiceAddedCapability
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables updates to the Broker Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Broker Service.

```
Get-BrokerServiceAddedCapability
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts

Get-BrokerServiceConfigurationData

March 11, 2024

View the Service configuration data objects.

Syntax

```
Get-BrokerServiceConfigurationData
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerServiceConfigurationData
    -SettingName <String>
    [-Property <String[]>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

View the Service configuration data objects stored in the broker database. These objects if set correctly, overrride the values present in the windows registry

————————————BrokerServiceConfigurationData Object

The service configuration object. It is a key value pair that can be used to tweek some settings in the broker

- SettingName (System.String)

  Name of the ServiceConfigurationData setting

- SettingValue (System.String)

  Value of the ServiceConfigurationData setting. Note that it need to be in the permissible range to correctly set it in the broker. The system will pickup defaults in all other cases.

Examples

EXAMPLE 1

Lists all the service configuration data configured by the admins in the broker

```
Get-BrokerServiceConfigurationData
```

Parameters

-SettingName

The canonical name of the service setting that can be overridden and stored in the database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32

Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ServiceConfigurationData

Returns the service settings set in the broker database

Related Links

- Set-BrokerServiceConfigurationData

Get-BrokerServiceInstance

March 11, 2024

Gets the service instance entries for the Broker Service.

Syntax

```
Get-BrokerServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Broker Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Broker Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
1  Get-BrokerServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Sdk.ServiceInstance

The Get-BrokerServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Broker.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Broker.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerServiceStatus
- Reset-BrokerServiceGroupMembership

Get-BrokerServiceStatus

March 11, 2024

Gets the current state of the Broker Service on the controller.

Syntax

```
Get-BrokerServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the Broker Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Broker Service.

```
Get-BrokerServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-BrokerServiceStatus command returns an object containing the status of the Broker Service together with extra diagnostics information.

DBUnconfigured

The Broker Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Broker Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Broker Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Broker is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Broker is connected to a database that is valid, but it does not have the full functionality required so the Broker cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Broker Service currently in use is incompatible with the version of the Broker Service schema on the database. Upgrade the Broker Service to a more recent version.

DBOlderVersionThanService

The version of the Broker Service schema on the database is incompatible with the version of the Broker Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Broker Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Broker Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Broker and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Broker service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Set-BrokerDBConnection
- Test-BrokerDBConnection
- Get-BrokerDBConnection
- Get-BrokerDBSchema

Get-BrokerSession

March 11, 2024

Gets a list of sessions.

Syntax

```
Get-BrokerSession
   [[-SessionKey] <Guid>]
   [-AgentVersion <String>]
   [-AnonymousUserId <String>]
   [-ApplicationInUse <String>]
   [-AppState <SessionAppState>]
   [-AppStateLastChangeTime <DateTime>]
   [-AutonomouslyBrokered <Boolean>]
   [-BrokeringDuration <Int32>]
```



```
    [-BrokeringTime <DateTime>]
    [-BrokeringUserName <String>]
    [-BrokeringUserSID <String>]
    [-CatalogName <String>]
    [-CbpVersion <CBPVersion>]
    [-ClientAddress <String>]
    [-ClientName <String>]
    [-ClientPlatform <String>]
    [-ClientProductId <Int32>]
    [-ClientVersion <String>]
    [-ConnectedViaHostName <String>]
    [-ConnectedViaIP <String>]
    [-ConnectionMode <ConnectionMode>]
    [-ControllerDNSName <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-DesktopKind <DesktopKind>]
    [-DesktopSID <String>]
    [-DesktopUid <Int32>]
    [-DeviceId <String>]
    [-DNSName <String>]
    [-EntitlementPolicyRuleUid <Int32>]
    [-EstablishmentDuration <Int32>]
    [-EstablishmentTime <DateTime>]
    [-HardwareId <String>]
    [-Hidden <Boolean>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionName <String>]
    [-IdleDuration <TimeSpan>]
    [-IdleSince <DateTime>]
    [-ImageOutOfDate <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAnonymousUser <Boolean>]
    [-IsPhysical <Boolean>]
    [-LaunchedViaHostName <String>]
    [-LaunchedViaIP <String>]
    [-LaunchedViaPublishedName <String>]
    [-LaunchedViaWorkspace <Boolean>]
    [-LogoffInProgress <Boolean>]
    [-LogonInProgress <Boolean>]
    [-MachineName <String>]
    [-MachineSummaryState <DesktopSummaryState>]
    [-MachineUid <Int32>]
    [-Metadata <String>]
    [-OSType <String>]
    [-PersistUserChanges <PersistUserChanges>]
    [-PowerState <PowerState>]
    [-PreferredZoneName <String>]
    [-PreferredZoneUid <Guid>]
    [-Protocol <String>]
    [-ProvisioningType <ProvisioningType>]
```



```
    [-ReceiverIPAddress <String>]
    [-ReceiverName <String>]
    [-SecureIcaActive <Boolean>]
    [-SessionId <Int32>]
    [-SessionReconnection <SessionReconnection>]
    [-SessionState <SessionState>]
    [-SessionStateChangeTime <DateTime>]
    [-SessionSupport <SessionSupport>]
    [-SessionType <SessionType>]
    [-StartTime <DateTime>]
    [-TenantId <Guid>]
    [-UntrustedUserName <String>]
    [-UserFullName <String>]
    [-UserName <String>]
    [-UserSID <String>]
    [-UserUPN <String>]
    [-ZoneName <String>]
    [-ZoneUid <Guid>]
    [-ApplicationUid <Int32>]
    [-SharedDesktopUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerSession
    [-Uid] <Int64>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves sessions matching all the specified criteria. If no parameters are specified this cmdlet enumerates all sessions.

————————BrokerSession Object

The session object returned represents a session on a machine in the site. The session could be for a desktop or application

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the machine.

- AnonymousUserId (System.String)

User ID associated with an anonymous session. This is a non-Windows identity and does not relate to the user account under which the session is running. Not all anonymous sessions have associated user IDs.

- ApplicationsInUse (System.String[])

  List of applications in use in the session. Applications are identified by their administrative name.

- AppState (Citrix.Broker.Admin.SDK.SessionAppState)

  The app state of the session. Valid values are PreLogon, PreLaunched, Active, Desktop, Lingering and NoApps.

- AppStateLastChangeTime (System.DateTime?)

  The time when the session entered the current app state.

- AutonomouslyBrokered (System.Boolean)

  Indicates whether this is an HDX session established by direct connection without being brokered.

- BrokeringDuration (System.Int32?)

  Time taken to broker the session (in milliseconds).

- BrokeringTime (System.DateTime?)

  Time at which the session was brokered.

- BrokeringUserName (System.String)

  The user name of the brokering user.

- BrokeringUserSID (System.String)

  The SID of the brokering user.

- CatalogName (System.String)

  The name of the catalog that the machine hosting the session is assigned to.

- CbpVersion (Citrix.Broker.Admin.SDK.CBPVersion?)

  The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

- ClientAddress (System.String)

  The IP address of the client connected to the session.

- ClientName (System.String)

  The host name of the client connected to the session.

- ClientPlatform (System.String)

  The name of client platform, as indicated by client product ID.

- ClientProductId (System.Int32?)

  The product ID of the client connected to the session.

- ClientVersion (System.String)

  The version of the Citrix Receiver running on the client connected to the session.

- ConnectedViaHostName (System.String)

  The host name of the incoming connection. This is usually a gateway, router or client.

- ConnectedViaIP (System.String)

  The IP address of the incoming connection This is usually a gateway, router or client.

- ConnectionMode (Citrix.Broker.Admin.SDK.ConnectionMode?)

  The way in which the most recent connection to the session was established. Valid modes are:

  - Brokered: established through the XenDesktop Broker (for example, using StoreFront).
  - Unbrokered: direct connection (for example, using the console or direct RDP/HDX).
  - LeasedConnection: established through the XenDesktop Broker using a connection lease.
  - VdaHighAvailabilityMode: direct connection while VDA in high-availability mode.
  - ThirdPartyBroker: established through a third-party Broker.
  - ThirdPartyBrokerWithLicensing: established and licensed through a third-party Broker.

- ControllerDNSName (System.String)

  The DNS host name of the controller that the session's hosting machine is registered with.

- DesktopGroupName (System.String)

  Name of the desktop group of the machine the session is on.

- DesktopGroupUid (System.Int32)

  UID of the desktop group of the machine the session is on.

- DesktopKind (Citrix.Broker.Admin.SDK.DesktopKind)

  Indicates if the session is shared or private.

- DesktopSID (System.String)

  The Windows SID of the machine the session is on.

- DesktopUid (System.Int32)

  For a desktop session, the unique identifier of the desktop.

- DeviceId (System.String)

  Unique identifier for the client device that has most recently been associated with the session.

- DNSName (System.String)

  The DNS host name of the machine hosting the session.

- EntitlementPolicyRuleUid (System.Int32?)

  The entitlement policy rule that granted the user the entitlement to launch the session. Only present for shared resources.

- EstablishmentDuration (System.Int32?)

  Duration that it took to establish the session.

- EstablishmentTime (System.DateTime?)

  Time at which the session was established.

- HardwareId (System.String)

  Unique identifier for the client hardware that has been most recently associated with the session.

- Hidden (System.Boolean)

  Flag to indicate if the session is currently hidden from the user and not to be reconnected to.

- HostedMachineName (System.String)

  The friendly name of a hosted machine running the session, as used by its hypervisor. This does not necessarily match either the DNS or AD name of the machine.

- HostingServerName (System.String)

  DNS name of the hypervisor that is hosting the machine hosting the session.

- HypervisorConnectionName (System.String)

  The name of the hypervisor connection that the machine hosting the session has been assigned to.

- IdleDuration (System.TimeSpan?)

  Period for which session has been idle, or null if it is not considered idle.

- IdleSince (System.DateTime?)

  Time at which session went idle

- ImageOutOfDate (System.Boolean?)

  Denotes whether the VM image for a hosted machine is out of date and due to be updated to a new master image when the machine next reboots.

- InMaintenanceMode (System.Boolean)

  Denotes whether the machine hosting the session is in maintentance mode.

- IPAddress (System.String)

  The IP address of the machine hosting the session.

- IsAnonymousUser (System.Boolean)

  Indicates whether the session was established anonymously (without user credentials), in this case a temporary local user account on the machine is used.

- IsPhysical (System.Boolean)

  This value is false if the machine hosting the session can be power managed, and true otherwise

- LaunchedViaHostName (System.String)

  The host name of the StoreFront server used to launch the session. This is blank if the session was launched via Workspace.

- LaunchedViaIP (System.String)

  The IP address of the StoreFront server used to launch the session. This is blank if the session was launched via Workspace.

- LaunchedViaPublishedName (System.String)

  The published name of the resource originally used to launch the session. This is the name selected by the user in Receiver. For application sessions this is the name of the first application launched into the session even if that application has since terminated. The name here does not change if the resource is later renamed or removed.

- LaunchedViaWorkspace (System.Boolean?)

  Indicates whether the session was launched via Citrix Workspace.

- LogoffInProgress (System.Boolean)

  Indicates whether the session is in the process of being logged off.

- LogonInProgress (System.Boolean)

  Indicates whether the session is still executing user logon processing or not.

- MachineName (System.String)

  DNS host name of the machine hosting the session.

- MachineSummaryState (Citrix.Broker.Admin.SDK.DesktopSummaryState)

  The summary state of the machine (will be Unregistered, Disconnected, or InUse)

- MachineUid (System.Int32)

  UID of the machine hosting the session.

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  Map of metadata for this session.

- OSType (System.String)

  A string that can be used to identify the operating system that is running on the machine hosting the session.

- PersistUserChanges (Citrix.Broker.Admin.SDK.PersistUserChanges)

  Describes whether/how the user changes are persisted. Possible values are:

  - OnLocal - Persist the user changes locally.
  - Discard - Discard user changes.

- PowerState (Citrix.Broker.Admin.SDK.PowerState)

  The current power state of the machine hosting the session. Possible values are: Unmanaged, Unknown, Unavailable, On, Suspended, TurningOn, TurningOff, Suspending and Resuming.

- PreferredZoneName (System.String)

  The name of preferred zone used for the original session launch. This may differ from the actual ZoneName if no resources were available in the preferred zone at launch time.

- PreferredZoneUid (System.Guid?)

  The UID of preferred zone used for the original session launch. This may differ from the actual ZoneUid if no resources were available in the preferred zone at launch time.

- Protocol (System.String)

  The protocol that the session is using, can be HDX, RDP, or Console. Console sessions on XenDesktop 5 VDAs appear with a blank protocol.

- ProvisioningType (Citrix.Broker.Admin.SDK.ProvisioningType)

  Describes how the machine hosting the session was provisioned, possible values are:

  - Manual: No provisioning.
  - PVS: Machine provisioned by PVS (may be physical, blade, VM,...)
  - MCS: Machine provisioned by MCS (machine must be VM)

- ReceiverIPAddress (System.String)

  The IP address of the client as supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

- ReceiverName (System.String)

  The name of the client as supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

- SecureIcaActive (System.Boolean?)

  Indicates whether SecureICA is active on the session.

- SessionId (System.Int32)

  Deprecated. A unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

- SessionKey (System.Guid)

  GUID that provides a unique identifier for this session.

- SessionReconnection (Citrix.Broker.Admin.SDK.SessionReconnection?)

  Describes the session reconnection (roaming) behavior for this session. Possible values are: Always, DisconnectedOnly, and SameEndpointOnly.

- SessionState (Citrix.Broker.Admin.SDK.SessionState)

  The state of the session. Valid values are Connected, Active or Disconnected.
  For a session on a machine with functional level below L7, the additional states PreparingSession, Reconnecting, NonBrokeredSession, Other, and Unknown can also occur.

- SessionStateChangeTime (System.DateTime)

  The time of the most recent state change for the session.

- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport)

  Indicates if the machine hosting the session supports multiple or single sessions.

- SessionType (Citrix.Broker.Admin.SDK.SessionType)

  Indicates if this is an Application or Desktop session.

- SmartAccessTags (System.String[])

  The Smart Access tags for this session.

- StartTime (System.DateTime?)

  Indicates when the session was started.

- TenantId (System.Guid?)

  The tenant associated with the session.

- Uid (System.Int64)

  Unique identifier of this session.

- UntrustedUserName (System.String)

  The name of the logged-on user reported directly from the machine. This may be useful where the user is logged in to a non-domain account, however the name cannot be verified and must be considered untrusted. The name is typically in the form DOMAIN\account, however in certain situations it may be reported as a UPN, that is, user@domain.

- UserFullName (System.String)

  The full name of the user.

- UserName (System.String)

  The name of the user.

- UserSID (System.String)

  The user's Windows SID.

- UserUPN (System.String)

  The user's User Principal Name

- ZoneName (System.String)

  The name of zone in which the machine hosting the session is located.

- ZoneUid (System.Guid)

  The UID of the zone in which the machine hosting the session is located.

Examples

EXAMPLE 1

Enumerate all sessions.

```
1  Get-BrokerSession
```

EXAMPLE 2

Get all disconnected sessions for a specific user.

```
1  Get-BrokerSession -UserName MyDomain\MyAccount -SessionState
       Disconnected
```


Parameters

-Uid

Get session by its Uid.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionKey

Gets session having the specified unique key.

Type:	Guid
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AgentVersion

Gets sessions with a specific Virtual Desktop Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-AnonymousUserId

Gets anonymous session associated with the specified user ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ApplicationInUse

Gets sessions running specific applications (identified by their SDK Name property).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AppState

Get sessions by their app state.

Valid values are PreLogon, PreLaunched, Active, Desktop, Lingering and NoApps.

Type:	SessionAppState
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppStateLastChangeTime

Get sessions by their app state change time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutonomouslyBrokered

Gets sessions according to whether they are autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringDuration

Gets session with a specific time taken to broker. In general, Citrix recommends using -Filter and relative comparisons.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringTime

Get sessions brokered at a specific time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringUserName

Get sessions by brokering user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-BrokeringUserSID

Get sessions by brokering user SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CatalogName

Gets sessions on machines from a specific catalog name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CbpVersion

The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

Type:	CBPVersion
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientAddress

Get sessions by client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientName

Get sessions by client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientPlatform

Get sessions by client platform.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ClientProductId

Get sessions by client product ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientVersion

Get sessions by client version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConnectedViaHostName

Get sessions by host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConnectedViaIP

Get sessions by IP address of the incoming connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ConnectionMode

Gets sessions by the way in which the most recent connection to the session was established.

Valid modes are Brokered, Unbrokered, LeasedConnection, VdaHighAvailabilityMode, ThirdPartyBroker, and ThirdPartyBrokerWithLicensing.

Type:	ConnectionMode
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets sessions that are hosted on machines which are registered with a specific controller.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-DesktopGroupName

Gets sessions from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets sessions from a desktop group with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Gets sessions on a desktop of a particular kind.

Valid values are Private and Shared.

Type:	DesktopKind
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopSID

Get sessions by desktop SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopUid

Get sessions by desktop Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeviceId

Get sessions by client device id.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DNSName

Gets sessions by their machine's DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-EntitlementPolicyRuleUid

Gets sessions where the user was granted the entitlement to launch the session from the specified entitlement policy rule.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EstablishmentDuration

Gets sessions which took a specific time to establish. In general, Citrix recommends using -Filter and relative comparisons.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EstablishmentTime

Gets sessions which became established at a particular time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HardwareId

Get sessions by client hardware id.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Hidden

Get sessions by whether they are hidden or not. Hidden sessions are treated as though they do not exist when brokering sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineName

Gets sessions by their machine's name as known to its hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingServerName

Gets sessions hosted by a machine with a specific name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-HypervisorConnectionName

Gets sessions hosted by a machine with a specific name of the hosting hypervisor connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IdleDuration

Gets sessions that have been idle for the specified period

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdleSince

Time at which session went idle

Type:	DateTime
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets sessions hosted by a machine with a specific ImageOutOfDate setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets sessions hosted by a machine with a specific InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets sessions hosted by a machine with a specific IP address.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsAnonymousUser

Gets sessions depending on whether they were established anonymously ($true) or not ($false). An anonymous session is established without user credentials and a temporary local user account is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPhysical

Gets sessions hosted on machines where the flag indicating if the machine can be power managed by the Citrix Broker Service matches the requested value. Where the power state of the machine cannot be controlled, specify $true, otherwise $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaHostName

Get sessions by the host name of the StoreFront server from which a user launches a session.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LaunchedViaIP

Get sessions by the IP address of the StoreFront server from which a user launches a session.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LaunchedViaPublishedName

Gets sessions originally launched using a resource having a name matching the specified published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LaunchedViaWorkspace

Gets sessions by whether they were launched via Citrix Workspace or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffInProgress

Gets sessions by whether they are in the process of being logged off or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogonInProgress

Gets sessions by whether they are still executing user logon processing or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets sessions by their machine name (in the form DOMAIN\machine).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineSummaryState

Gets sessions on a machine with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, Preparing, and InUse.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets sessions on a machine with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets sessions with a specific type of operating system.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PersistUserChanges

Gets sessions where the user changes are persisted in a particular manner. Values can be:

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets sessions on machines in the specified power state.

Valid values are Unmanaged, Unknown, Unavailable, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredZoneName

Gets sessions originally launched with the specified preferred zone name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PreferredZoneUid

Gets sessions originally launched with the specified preferred zone Uid.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Protocol

Get sessions by connection protocol. Valid values are HDX, RDP, or Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningType

Gets sessions hosted on machines provisioned in a particular manner. Values can be:

- Manual - No automated provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ReceiverIPAddress

Gets sessions with the specified client IP address supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReceiverName

Gets sessions with the specified client name supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SecureIcaActive

Get sessions by their use of SecureICA.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionId

Deprecated.

Gets sessions by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Get sessions by their session reconnection (roaming) behavior. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Get sessions by their state.

Valid values are Other, PreparingNewSession, Connected, Active, Disconnected, Reconnecting, Non-BrokeredSession, and Unknown.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Get sessions by their last state change time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSupport

Gets sessions hosted on machines which support the required pattern of sessions. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionType

Get sessions by their type.

Valid values are Application and Desktop.

Type:	SessionType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Get sessions by their start time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Gets sessions associated with the specified tenant.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UntrustedUserName

Gets sessions by the untrusted user name reported directly from the machine.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserFullName

Gets sessions by user's full name (usually 'first-name last-name').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserName

Get sessions by user name (in the form DOMAIN\user).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserSID

Get sessions by user's Windows SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserUPN

Gets sessions by user's User Principal Name (in the form user@domain).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneName

Gets sessions hosted on machines located in the zone with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets sessions hosted on machines located in the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Get sessions running the application with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SharedDesktopUid

Get sessions by SharedDesktop Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Session

Returns sessions matching the specified criteria.

Related Links

- Disconnect-BrokerSession
- Stop-BrokerSession
- Get-BrokerDesktop
- about_Broker_Filtering

Get-BrokerSessionLinger

March 11, 2024

Gets one or more session lingering settings.

Syntax

```
Get-BrokerSessionLinger
    [[-DesktopGroupName] <String>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserSID <String>]
    [-AssociatedUserUPN <String>]
```



```
    [-Enabled <Boolean>]
    [-MaxAverageLoadThreshold <Int32>]
    [-MaxLoadPerMachineThreshold <Int32>]
    [-MaxTimeBeforeDisconnect <TimeSpan>]
    [-MaxTimeBeforeTerminate <TimeSpan>]
    [-UserFilterEnabled <Boolean>]
    [-UserSID <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerSessionLinger
    [-DesktopGroupUid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerSessionLinger cmdlet is used to enumerate desktop group session linger settings that match all of the supplied criteria.

Without parameters, Get-BrokerSessionLinger gets all the session linger settings that have been created. You can also use the parameters of Get-BrokerSessionLinger to filter the results to just the desktop group you're interested in.

Note that each desktop group can only have a single session linger setting. Session lingering only applies to application sessions.

See about_Broker_Filtering for information about advanced filtering options.

————————BrokerSessionLinger Object

The session linger object returned represents a session linger setting in a desktop group.

- AssociatedUserFullNames (System.String[])

  List of associated users (full names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserNames (System.String[])

  List of associated users (SAM names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserSIDs (System.String[])

  List of associated users (SIDs). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserUPNs (System.String[])

  List of associated users (user principle names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- DesktopGroupName (System.String)

  Name of the associated desktop group.

- DesktopGroupUid (System.Int32)

  Uid of the associated desktop group.

- Enabled (System.Boolean)

  Specifies whether or not session lingering is enabled for the desktop group.

- MaxAverageLoadThreshold (System.Int32)

  Specifies the average load threshold across the desktop group. After this threshold is hit lingering sessions will be terminated to reduce average load across the group. Sessions that have been lingering the longest will be chosen first.

- MaxLoadPerMachineThreshold (System.Int32)

  Specifies the maximum load threshold per machine in the desktop group. After this threshold is hit lingering sessions on loaded machines will be terminated to reduce load. Sessions that have been lingering the longest will be chosen first.

- MaxTimeBeforeDisconnect (System.TimeSpan)

  Specifies the maximum time by when a lingering session will be disconnected. The disconnect timer cannot be greater than the terminate timer. When the disconnect timer is same as the terminate timer, the session will be directly be terminated. The default value is 15 minutes. A value of 0 disables the disconnect timer.

- MaxTimeBeforeTerminate (System.TimeSpan)

  Specifies the maximum time by when a lingering session will be terminated. When the disconnect timer is same as the terminate timer, the session will be directly be terminated. The default value is 8 hours. A value of 0 disables the terminate timer.

- UserFilterEnabled (System.Boolean)

  Indicates if linger-specific user filter is enabled.

Examples

EXAMPLE 1

Returns the session linger settings associated with the destkop group named 'test'.

```
Get-BrokerSessionLinger -DesktopGroupName "test"
```

Parameters

-DesktopGroupUid

Gets session linger setting that is associated with the specified desktop group Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets session linger setting that is associated with the specified desktop group name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserFullName

Gets session linger settings with an associated user identified by their full name (usually 'first-name last-name'). If the 'UserFilterEnabled'property is true then access to the session linger is restricted to

those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets session linger settings with an associated user identified by their user name (in the form 'domain\user'). If the 'UserFilterEnabled'property is true then access to the session linger is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserSID

Gets session linger settings with an associated user identified by their Windows SID. If the 'UserFilterEnabled'property is true then access to the session linger is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-AssociatedUserUPN

Gets session linger settings with an associated user identified by their user principle name (in the form 'user@domain'). If the 'UserFilterEnabled'property is true then access to the session linger is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Enabled

Gets only the session linger settings that have the specified value for whether the setting is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Gets only the session linger settings that have the specified average load threshold.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Gets only the session linger settings that have the specified maximum load threshold per machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Gets only the session linger settings that have the specified idle disconnect time.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Gets only the session linger settings that have the specified idle terminate time.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Gets only session linger settings whose user filter is in the specified state.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserSID

Gets only session linger settings with their accessibility restricted to include the specified user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.SessionLinger

Get-BrokerSessionLinger returns an object for each session linger setting it gets.

Related Links

- New-BrokerSessionLinger
- Set-BrokerSessionLinger
- Remove-BrokerSessionLinger

Get-BrokerSessionPreLaunch

March 11, 2024

Gets one or more session pre-launch settings.

Syntax

```
Get-BrokerSessionPreLaunch
   [[-DesktopGroupName] <String>]
   [-AssociatedUserFullName <String>]
   [-AssociatedUserName <String>]
   [-AssociatedUserSID <String>]
   [-AssociatedUserUPN <String>]
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-UserSID <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
```



```
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerSessionPreLaunch
    [-DesktopGroupUid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerSessionPreLaunch cmdlet is used to enumerate desktop group session pre-launch settings that match all of the supplied criteria.

Without parameters, Get-BrokerSessionPreLaunch gets all the session pre-launch settings that have been created. You can also use the parameters of Get-BrokerSessionPreLaunch to filter the results to just the desktop group you're interested in.

Note that each desktop group can only have a single session pre-launch setting. Session pre-launch only applies to application sessions.

See about_Broker_Filtering for information about advanced filtering options.

————————BrokerSessionPreLaunch Object

The session pre-launch object returned represents a session pre-launch setting in a desktop group.

- AssociatedUserFullNames (System.String[])

  List of associated users (full names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserNames (System.String[])

  List of associated users (SAM names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserSIDs (System.String[])

  List of associated users (SIDs). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- AssociatedUserUPNs (System.String[])

  List of associated users (user principle names). Associated users is the list of users who are given access using the pre-launch/user mapping filter.

- DesktopGroupName (System.String)

  Name of the associated desktop group.

- DesktopGroupUid (System.Int32)

  Uid of the associated desktop group.

- Enabled (System.Boolean)

  Specifies whether or not session pre-launch is enabled for the desktop group.

- MaxAverageLoadThreshold (System.Int32)

  Specifies the average load threshold across the desktop group. After this threshold is hit pre-launched sessions will be terminated to reduce average load across the group. Sessions that have been pre-launched the longest will be chosen first.

- MaxLoadPerMachineThreshold (System.Int32)

  Specifies the maximum load threshold per machine in the desktop group. After this threshold is hit pre-launched sessions on loaded machines will be terminated to reduce load. Sessions that have been pre-launched the longest will be chosen first.

- MaxTimeBeforeDisconnect (System.TimeSpan)

  Specifies the maximum time by when a pre-launched session will be disconnected. The disconnect timer cannot be greater than the terminate timer. When the disconnect timer is same as the terminate timer, the session will be directly be terminated. The default value is 15 minutes. A value of 0 disables the disconnect timer.

- MaxTimeBeforeTerminate (System.TimeSpan)

  Specifies the maximum time by when a pre-launched session will be terminated. When the disconnect timer is same as the terminate timer, the session will be directly be terminated. The default value is 8 hours. A value of 0 disables the terminate timer.

- UserFilterEnabled (System.Boolean)

  Indicates if pre-launch-specific user filter is enabled.

Examples

EXAMPLE 1

Returns the session pre-launch settings associated with the destkop group named ‘test’.

```
1  Get-BrokerSessionPreLaunch -DesktopGroupName "test"
```


Parameters

-DesktopGroupUid

Gets session pre-launch setting that is associated with the specified desktop group Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets session pre-launch setting that is associated with the specified desktop group name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserFullName

Gets session pre-launch settings with an associated user identified by their full name (usually 'first-name last-name'). If the 'UserFilterEnabled'property is true then access to the session pre-launch is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserName

Gets session pre-launch settings with an associated user identified by their user name (in the form 'domain\user'). If the 'UserFilterEnabled'property is true then access to the session pre-launch is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserSID

Gets session pre-launch settings with an associated user identified by their Windows SID. If the 'User-FilterEnabled'property is true then access to the session pre-launch is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AssociatedUserUPN

Gets session pre-launch settings with an associated user identified by their user principle name (in the form 'user@domain'). If the 'UserFilterEnabled'property is true then access to the session pre-launch is restricted to those users only, otherwise access is unrestricted (but always subject to other policy rules).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Enabled

Gets only the session pre-launch settings that have the specified value for whether the setting is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Gets only the session pre-launch settings that have the specified average load threshold.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Gets only the session pre-launch settings that have the specified maximum load threshold per machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Gets only the session pre-launch settings that have the specified idle disconnect time.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Gets only the session pre-launch settings that have the specified idle terminate time.

Type:	TimeSpan
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Gets only session pre-launch settings whose user filter is in the specified state.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserSID

Gets only session pre-launch settings with their accessibility restricted to include the specified user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.SessionPreLaunch

Get-BrokerSessionPreLaunch returns an object for each session pre-launch setting it gets.

Related Links

- New-BrokerSessionPreLaunch
- Set-BrokerSessionPreLaunch
- Remove-BrokerSessionPreLaunch

Get-BrokerSessionRecordingStatus

March 11, 2024

Gets the recording status of the specified session.

Syntax

```
Get-BrokerSessionRecordingStatus
   [-Session] <Session>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the recording status of the specified session.

Examples

EXAMPLE 1

Gets the recording status of the session Uid 1.

```
Get-BrokerSessionRecordingStatus -Session 1
```

Parameters

-Session

The session whose recording status to get.

Type:	Session
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The session whose recording status to get.

Outputs

String

Get-BrokerSessionRecordingStatus returns a string representing whether or not the target session is being recorded.

Values can be:

- SessionBeingRecorded - Session is being recorded.
- SessionNotRecorded - Session is not being recorded.

Notes

The specified session can be passed using the Session parameter as a session object or a session Uid.

Related Links

- Start-BrokerSessionRecording
- Stop-BrokerSessionRecording

Get-BrokerSharedDesktop

March 11, 2024

Get shared desktops configured for this site.

Syntax


```
Get-BrokerSharedDesktop
    [[-MachineName] <String>]
    [-AgentVersion <String>]
    [-ControllerDNSName <String>]
    [-DesktopGroupUid <Int32>]
    [-DNSName <String>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-LastDeregistrationReason <DeregistrationReason>]
    [-LastDeregistrationTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-PowerState <PowerState>]
    [-RegistrationState <RegistrationState>]
    [-SID <String>]
    [-Tag <String>]
    [-WillShutdownAfterUse <Boolean>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerSharedDesktop
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet is deprecated, please use the Get-BrokerMachine cmdlet instead.

Retrieve shared desktops matching the specified criteria. If no parameters are specified, all shared desktops are enumerated.

Get-BrokerSharedDesktop returns configuration information only for shared desktops (a DesktopKind of 'Shared').

For information about advanced filtering options, see about_Broker_Filtering; for information about desktops, see about_Broker_Desktops.

——————————BrokerSharedDesktop Object

Shared desktops are desktops that are assigned randomly to users upon connection from a pool of available machines.

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the desktop.

- ControllerDNSName (System.String)

  The DNS host name of the controller that the desktop is registered to.

- DesktopGroupUid (System.Int32)

  Uid of the desktop group the desktop has been assigned to.

- DNSName (System.String)

  The DNS host name of the desktop.

- HostedMachineId (System.String)

  Unique ID within the hosting unit of the target managed desktop.

- HostedMachineName (System.String)

  The friendly name of a hosted desktop as used by its hypervisor. This is not necessarily the DNS name of the desktop.

- HostingServerName (System.String)

  DNS name of the hypervisor that is hosting the desktop if managed.

- HypervisorConnectionUid (System.Int32?)

  The UID of the hypervisor connection that the desktop has been assigned to, if managed.

- InMaintenanceMode (System.Boolean)

  Denotes whether the desktop is in maintentance mode.

- IPAddress (System.String)

  The IP address of the desktop.

- LastDeregistrationReason (Citrix.Broker.Admin.SDK.DeregistrationReason?)

  The reason for the last deregistration of the desktop with the broker. Possible values are: AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog,

FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

- LastDeregistrationTime (System.DateTime?)

  Time of the last deregistration of the desktop from the controller.

- LastHostingUpdateTime (System.DateTime?)

  Time of last update to any hosting data for this desktop reported by the hypervisor connection.

- MachineName (System.String)

  DNS host name of the machine associated with the desktop.

- OSType (System.String)

  A string that can be used to identify the operating system that is running on the desktop.

- OSVersion (System.String)

  A string that can be used to identify the version of the operating system running on the desktop, if known.

- PowerState (Citrix.Broker.Admin.SDK.PowerState)

  The current power state of the desktop. Possible values are: Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, resuming.

- RegistrationState (Citrix.Broker.Admin.SDK.RegistrationState)

  Indicates the registration state of the desktop. Possible values are: Unregistered, Initializing, Registered, AgentError.

- SID (System.String)

  The security identifier of the shared desktop.

- Uid (System.Int32)

  The uid of the shared desktop.

- WillShutdownAfterUse (System.Boolean)

  Flag indicating whether this desktop is tainted and will be shutdown after all sessions on the desktop have ended. This flag should only ever be true on power managed, single-session desktops.
  Note: The desktop will not shut down if it is in maintenance mode, but will shut down after the desktop is taken out of maintenance mode.

Examples

EXAMPLE 1

Get all shared desktops hosted by the hypervisor server BigServer12.

```
Get-BrokerSharedDesktop -HostingServerName BigServer12*
```

EXAMPLE 2

List all shared desktops running Windows XP, including the machine name and OS details.

```
Get-BrokerSharedDesktop -OSType 'Windows XP*' | ft -a MachineName,
    OSType,OSVersion
```

Parameters

-Uid

Gets desktops by Uid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets desktops by machine name (in the form 'domain\machine').

Type:	String
Position:	2
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-AgentVersion

Gets desktops with a specific Virtual Desktop Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ControllerDNSName

Gets desktops by the DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DesktopGroupUid

Gets desktops from a desktop group with a specific Uid.

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets desktops by DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineId

Gets desktops by the machine id known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostedMachineName

Gets desktops by the machine name known to the hypervisor.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingServerName

Gets desktops by the name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HypervisorConnectionUid

Gets desktops by the uid of the hosting hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets desktops by the InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets desktops by IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LastDeregistrationReason

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationTime

Gets desktops by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets desktops by the time that the hosting information was last updated.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets desktops by the type of operating system they are running.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets desktops by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PowerState

Gets desktops by power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RegistrationState

Gets desktops by registration state.

Valid values are Registered, Unregistered, and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Gets desktops by machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Tag

Get desktops tagged with the given tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-WillShutdownAfterUse

Gets desktops depending on whether they shutdown after use or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.SharedDesktop

Get-BrokerSharedDesktop returns an object for each matching shared desktop.

Notes

To compare dates/times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering.

Related Links

- about_Broker_Filtering
- about_Broker_Desktops
- Set-BrokerSharedDesktop

Get-BrokerSite

March 11, 2024

Gets the current XenDesktop broker site.

Syntax

```
Get-BrokerSite
    [-ReuseMachinesWithoutShutdownInOutageAllowed <Boolean>]
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-BrokerSite cmdlet gets the current broker site.

The broker site is a top-level, logical representation of the XenDesktop site, from the perspective of the brokering services running within the site. It defines various site-wide default attributes used by the brokering services.

A XenDesktop installation has only a single broker site instance.

————————BrokerSite Object

The BrokerSite object represents logical representation of the XenDesktop site. It contains the following properties:

- AlwaysBypassAuthForCachedResources (System.Boolean)

  Allows client to always display cached resources without authentication.

- ApplicationIconUid (System.Int32)

  The default icon used for new applications if one is not explicitly specified.

- BaseOU (System.Guid?)

  The objectGUID property identifying the base OU in Active Directory used for desktop registrations.

- BrokerServiceGroupUid (System.Guid)

  The Uid for the Broker Service Group.

- BypassAuthForCachedResources (System.Boolean)

  Allows client to display cached resources without authentication.

- CloudSiteLicense (System.String)

  Configures the single cloud license chosen to be used as the default one for the site.

- CloudValidLicenses (System.String)

  The valid cloud license SKUs.

- ColorDepth (Citrix.Broker.Admin.SDK.ColorDepth)

  The default color depth for new desktop groups.

- ConfigLastChangeTable (System.String)

  The last table that was changed as part of the last change to the broker configuration.

- ConfigLastChangeTime (System.DateTime)

  The time the broker configuration was changed.

- ConfigurationServiceGroupUid (System.Guid?)

  The Uid for the Configuration Service Group.

- ConnectionLeasingEnabled (System.Boolean?)

  Always false. Connection leasing is no longer supported.

- CredentialForwardingToCloudAllowed (System.Boolean)

  The indicator that whether the Connector is allowed to forward user credentials to cloud.

- DefaultMinimumFunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel?)

  The default minimum functional level used for new catalogs and desktop groups when no explicit value is provided.

- DefaultReuseMachinesWithoutShutdownInOutage (System.Boolean)

  The default ReuseMachinesWithoutShutdownInOutage used for new desktop groups when no explicit value is provided.

- DeleteResourceLeasesOnLogOff (System.Boolean)

  Forces client to delete all leases on explicit logoff.

- DesktopGroupIconUid (System.Int32)

  The default icon used for new desktop groups if one is not explicitly specified.

- DnsResolutionEnabled (System.Boolean)

  The setting to configure whether numeric IP address or the DNS name to be present in the ICA file.

- GpoCustomTemplatesGuid (System.Guid)

  The Guid of the GPO policy set that stores the policy templates defined by the site administrator.

- GpoPoliciesGuid (System.Guid)

  The Guid of the GPO policy set that stores the policies for the site.

- GpoTemplatesGuid (System.Guid)

  The Guid of the GPO policy set that stores the policy templates for the site.
- InMemorySchemaAppliedVersion (System.Int32)

  Version of in-memory SQL table optimization applied to database schema.
- InMemorySchemaSupportedVersion (System.Int32)

  Version of in-memory SQL table optimization supported by database schema.
- IsSecondaryBroker (System.Boolean)

  Reserved for internal use.
- LicensedSessionsActive (System.Int32?)

  The count of active licensed session.
- LicenseEdition (System.String)

  The license edition for session brokering.
- LicenseGraceSessionsRemaining (System.Int32?)

  The count of Concurrent License Grace Sessions Remaining
- LicenseModel (Citrix.Broker.Admin.SDK.LicenseModel?)

  The licensing model in use. Values can be 'Concurrent' or 'UserDevice'
- LicenseServerName (System.String)

  The DNS for License Server Name
- LicenseServerPort (System.Int32)

  The port for the License Server
- LicensingBurnIn (System.String)

  The date for the license to end in yyyy.MMdd format
- LicensingBurnInDate (System.DateTime?)

  The date for the license to end
- LicensingGraceHoursLeft (System.Int32?)

  The number of grace hours left after license expiry
- LicensingGracePeriodActive (System.Boolean?)

  The indicator for licensing grace period active
- LicensingOutOfBoxGracePeriodActive (System.Boolean?)

  The indicator for licensing out of the box grace period active

- LocalHostCacheEnabled (System.Boolean)

  The indicator that the Local Host Cache feature is switched on

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  The metadata for this command.

- Name (System.String)

  The name of the site

- PeakConcurrentLicensedDevices (System.Int32?)

  The peak number of concurrent devices

- PeakConcurrentLicenseUsers (System.Int32?)

  The peak number of concurrent license users

- PreferredAccountName (System.String)

  Determines if SAM name or UPN should be displayed for default name of user/group account

- RequireXmlServiceKeyForNFuse (System.Boolean)

  Determines if the NFuse, MCP, and Admin interfaces require authentication with a service key.

- RequireXmlServiceKeyForSta (System.Boolean)

  Determines if the StA interface requires authentication with a service key.

- ResourceLeaseValidityPeriodInDays (System.Int32)

  Validity period for a lease.

- ResourceLeasingEnabled (System.Boolean)

  Enables lease syncing on client.

- ReuseMachinesWithoutShutdownInOutageAllowed (System.Boolean)

  Specifies whether or not power cycle behavior during outage can be overriden on a delivery group level.

- SecureIcaRequired (System.Boolean)

  The default SecureICA usage requirements for new desktop groups.

- TelemetryHeadlessLaunchEnabled (System.Boolean)

  Enables client to perform headless telemetry launches.

- TelemetryLaunchMinTimeIntervalMins (System.Int32)

  Configures minimum time interval (in minutes) between headless telemetry launches.

- TelemetryLaunchShadowDelayMins (System.Int32)

  Configures delay (in minutes) between ICA-HDX launch and headless telemetry launch.

- TotalUniqueLicenseUsers (System.Int32?)

  The total count of license users

- TrustManagedAnonymousXmlServiceRequests (System.Boolean)

  The XML Service managed anonymous settings.

- TrustRequestsSentToTheXmlServicePort (System.Boolean)

  The XML Service trust settings.

- UseVerticalScalingForRdsLaunches (System.Boolean)

  Use vertical scaling when finding an RDS machine for a session launch.

- XmlServiceKey1 (System.String)

  The first XML Service Key

- XmlServiceKey2 (System.String)

  The second XML Service Key

Examples

EXAMPLE 1

Gets the current broker site.

```
Get-BrokerSite
```

Parameters

-ReuseMachinesWithoutShutdownInOutageAllowed

Specifies whether or not power cycle behavior during outage can be overriden on a delivery group level.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Site

Get-BrokerSite returns the single broker site instance.

Related Links

- about_Broker_Concepts
- Set-BrokerSite
- Get-BrokerIcon

Get-BrokerStorefrontAddress

March 11, 2024

Gets the high-level description of a configuration for StoreFront addresses, based on a configuration byte array.

Syntax

```
Get-BrokerStorefrontAddress
   [-ByteArray] <Byte[]>
   [<CommonParameters>]
```

Description

Use this command to convert a configuration byte array into a set of named property settings. The byte array will either have been retrieved from the Citrix Broker Service, or from the New-BrokerStorefrontAddress cmdlet.

Examples

EXAMPLE 1

This example shows a new configuration byte array being created to specify a single StoreFront address. The configuration byte array is then provided as input to the Get-BrokerStorefrontAddress command, which interprets and outputs the same fields.


```
$configuration = New-BrokerStorefrontAddress -Url "https://mysite.com/
    Citrix/StoreWeb" -Description "This StoreFront delivers my corporate
     applications" -Name "StoreFront1" -Enabled $true

Get-BrokerStorefrontAddress -ByteArray $configuration

Name               Url                                        Enabled
    Description
----               ---                                        -------
    -----------
StoreFront1        https://mysite.com/Citrix/StoreWeb           True
    This StoreFront delivers my corporate applications.
```

Parameters

-ByteArray

Specifies the low-level byte array (blob) to be interpreted.

Type:	Byte[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Byte[]

The cmdlet accepts the ByteArray parameter as pipeline input.

Outputs

Citrix.Storefront.Sdk.BrokerStorefrontAddress

This cmdlet outputs one BrokerStorefrontAddress object for each address that is configured within the slot. Each object has the following properties:

Name - Specifies the name of the StoreFront.

Url - Specifies the URL to the StoreFront, such as "https://mysite.com/Citrix/StoreWeb".

Enabled - Specifies whether the StoreFront is enabled for users.

Description - Specifies the human-readable name of the StoreFront.

Related Links

- New-BrokerStorefrontAddress
- Add-BrokerStorefrontAddress
- New-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration

Get-BrokerTag

March 11, 2024

Gets one or more tags.

Syntax

```
Get-BrokerTag
   [[-Name] <String>]
   [-Description <String>]
   [-IsUsedByGpo <Boolean>]
   [-Metadata <String>]
   [-ScopeId <Guid>]
   [-ScopeName <String>]
   [-UUID <Guid>]
   [-ApplicationUid <Int32>]
   [-ApplicationGroupUid <Int32>]
   [-CatalogUid <Int32>]
   [-DesktopUid <Int32>]
   [-DesktopGroupUid <Int32>]
```



```
    [-MachineUid <Int32>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerTag
    [-Uid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets tags that match all of the supplied criteria.

————————BrokerTag Object

The BrokerTag object represents a single instance of a Tag associated to other objects. It contains the following properties:

- Description (System.String)

  Description of the tag

- IsUsedByGpo (System.Boolean?)

  Indicate if the tag is used by group policy

- MetadataMap (System.Collections.Generic.Dictionary<string, string>)

  The metadata for this command.

- Name (System.String)

  The name of the Tag

- Scopes (Citrix.Fma.Sdk.DelegatedAdminInterfaces.ScopeReference[])

  The list of the delegated admin scopes to which the tag belongs.

- Uid (System.Int32)

  The Uid of the Tag

- UUID (System.Guid)

  The UUID associated to the Tag

Examples

EXAMPLE 1

Enumerate all tags.

```
Get-BrokerTag
```

EXAMPLE 2

Get a single specific tag with a Uid of 5.

```
Get-BrokerTag -Uid 5
```

EXAMPLE 3

Get tags associated with machine DOMAIN\Machine.

```
$machine = Get-BrokerMachine DOMAIN\Machine
Get-BrokerTag -MachineUid $machine.Uid
```

Parameters

-Uid

Gets the tag identified by Uid

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

Gets tags that match the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets tags with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsUsedByGpo

Gets tags that are used by group policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeId

Gets tags that are associated with the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScopeName

Gets tags that are associated with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UUID

Gets tags associated with a given UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets tags associated with the specified application.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationGroupUid

Get tags associated with the specified application group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUid

Gets tags associated with the specified catalog.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopUid

Gets tags associated with the specified desktop.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets tags associated with the specified desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets tags associated with the specified machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Tag

Returns matching tags.

Related Links

- Add-BrokerTag
- New-BrokerTag
- Remove-BrokerTag
- Rename-BrokerTag
- Set-BrokerTag
- about_Broker_Filtering

Get-BrokerTagUsage

March 11, 2024

Produces a usage report for one or more tags.

Syntax


```
Get-BrokerTagUsage
    [[-TagName] <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerTagUsage
    [-TagUid] <Int32>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns one BrokerTagUsage for each input tag. Each BrokerTagUsage object summarizes the tag's usage throughout the site.

————————BrokerTagUsage Object

Each BrokerTagUsage object summarizes on the usage of a single Tag.

- TaggedApplicationGroups (System.Int32)

  The number of application groups that are tagged with this tag and that are visible to the calling delegated administrator.

- TaggedApplications (System.Int32)

  The number of applications that are tagged with this tag and that are visible to the calling delegated administrator.

- TaggedCatalogs (System.Int32)

  The number of catalogs that are tagged with this tag and that are visible to the calling delegated administrator.

- TaggedDesktopGroups (System.Int32)

  The number of desktop groups that are tagged with this tag and that are visible to the calling delegated administrator.

- TaggedMachines (System.Int32)

  The number of machines that are tagged with this tag and that are visible to the calling delegated administrator.

- TagName (System.String)

  The name of the Tag.

- TagRestrictedApplicationGroups (System.Int32)

  The number of application groups that have this tag restriction and that are visible to the calling delegated administrator.

- TagRestrictedAutoscale (System.Int32)

  The number of desktop groups that have this tag restriction

- TagRestrictedEntitlementPolicyRules (System.Int32)

  The number of desktop rules in the site's entitlement policy that have this tag restriction and that are visible to the calling delegated administrator.

- TagRestrictedRebootSchedules (System.Int32)

  The number of reboot schedules that have this tag restriction.

- TagUid (System.Int32)

  The Uid of the Tag.

- TagUUID (System.Guid)

  The UUID associated to the Tag.

- TotalTaggedObjects (System.Int32)

  The total number of objects that are tagged with this tag.

- TotalTagRestrictedObjects (System.Int32)

  The total number of objects that have this tag restriction.

- UnknownTaggedObjects (System.Int32)

  The number of objects of all types that are tagged with this tag and that are *not* visible to the calling delegated administrator.

- UnknownTagRestrictedObjects (System.Int32)

  The number of objects of all types that have this tag restriction and that are *not* visible to the calling delegated administrator.

Examples

EXAMPLE 1

Reports on the usage of all tags in the site.


```
Get-BrokerTagUsage
```

EXAMPLE 2

Reports on the usage of the tag MyTag.

```
Get-BrokerTagUsage MyTag
```

Parameters

-TagUid

Report on the tag identified by TagUid.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TagName

Report on the tag identified by TagName.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.TagUsage

Summarizes the usage of a single tag.

Notes

If a tag is applied to an object, but that object would not be visible to the calling delegated administrator via the corresponding Get-Broker* cmdlet, then the object is not counted as an object of that type, but is instead reported as an 'unknown' object.

For example, suppose that the caller has invoked Get-BrokerTagUsage MyTag, and that there are three desktop groups and no other objects tagged with MyTag. Two of the desktop groups are visible to the caller, but the other is not. In this case, TotalTaggedObjects is 3, TaggedDesktopGroups is 2, and UnknownTaggedObjects is 1.

Related Links

- Get-BrokerTag

Get-BrokerTelemetryData

March 11, 2024

Generates list of registered instances of Public cloud workloads

Syntax

```
Get-BrokerTelemetryData
    [-RegistrationTime <DateTime>]
    [-SID <String>]
    [-Uid <Int64>]
```



```
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns list of Public cloud workloads and their telemetry usage metadata

————————BrokerTelemetryData Object

Data objects for filtering

- Instance (System.String)

  InstanceMetadata Object

- InstanceType (System.String)

  Instance type of the machine provided in a AWS instance

- Location (System.String)

  Location of the hosted machine

- MachineType (System.String)

  Machine type of the instance in a GCP machine

- ProjectId (System.String)

  Project Id of the account in GCP project

- RegistrationTime (System.DateTime?)

  Actual Registration Time of the VDA

- ResourceGroupName (System.String)

  Resource Group name of the resource in an Azure environment

- ResourceId (System.String)

  Resource ID of the resource in Azure environment

- SID (System.String)

  SId of the instance

- SubscriptionId (System.String)

  Subscription record for Azure

- Timestamp (System.DateTime?)

  Metadata timestamp received from the VDA

- Uid (System.Int64)

  Uid of the instance metadata

- VmId (System.String)

  VmId of the instance

- VmSize (System.String)

  VM size of the Azure instance

Examples

EXAMPLE 1

Generates list of all registered VDA for public cloud workloads

```
Get-BrokerTelemetryData
```

Parameters

-RegistrationTime

VDA Registration time of the service instance

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Gets machines with a specific machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Uid

Id of the registered VDA instance

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String

Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet

Outputs

Citrix.Broker.Admin.SDK.TelemetryData

Notes

None

None

Related Links

- Get-BrokerTelemetryData

Get-BrokerUnconfiguredMachine

March 11, 2024

Gets machines that have registered but are not yet configured in this site.

Syntax

```
Get-BrokerUnconfiguredMachine
    [[-MachineName] <String>]
    [-AgentVersion <String>]
    [-ControllerDNSName <String>]
    [-CurrentlyRegistered <Boolean>]
    [-DNSName <String>]
    [-FunctionalLevel <FunctionalLevel>]
    [-LastDeregistrationTime <DateTime>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-SessionSupport <SessionSupport>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerUnconfiguredMachine
    -SID <String>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve machines matching the specified criteria, where the machine has or was previously registered with a controller in the site, but the machine has not yet been configured to be part of the site. If no parameters are specified, this cmdlet enumerates all such machines that are currently registered.

See about_Broker_Filtering for information about advanced filtering options, and about_Broker_Machines for background information about machines.

——————————BrokerUnconfiguredMachine Object

An unconfigured machine is a machine that has registered with the site but is not configured in either a desktop group or a catalog.

- AgentVersion (System.String)

  Version of the Citrix Virtual Delivery Agent (VDA) installed on the unconfigured machine.

- ControllerDNSName (System.String)

  The DNS name of the controller that the unconfigured machine is registered with. This property's value is null for machines that are no longer registered.

- CurrentlyRegistered (System.Boolean)

  True if the unconfigured machine is currently registered with the site, or false if the unconfigured machine is no longer registered with the site (but was registered at some point in the past).

- DNSName (System.String)

  The DNS name of the unconfigured machine.

- FunctionalLevel (Citrix.Broker.Admin.SDK.FunctionalLevel?)

  The functional level of the unconfigured machine. This is determined by the version of the Citrix VDA software installed on the machine.

- LastDeregistrationTime (System.DateTime?)

  The time when the unconfigured machine last deregistered with the Citrix Broker Service.

- MachineName (System.String)

  The machine name of the unconfigured machine in the form domain\machine.

- OSType (System.String)

  The type of operating system installed on the unconfigured machine.

- OSVersion (System.String)

  The version of the operating sysetem installed on the unconfigured machine.

- SessionSupport (Citrix.Broker.Admin.SDK.SessionSupport?)

  The session support of the unconfigured machine. Valid values are:
  SingleSession, MultiSession

- SID (System.String)

  Security identifier of the unconfigured machine.

Examples

EXAMPLE 1

This command returns all unconfigured XP SP3 machines which are registered with the controller called 'Controller1.domain.com'.

```
Get-BrokerUnconfiguredMachine -Filter {
  ControllerDNSName -eq 'Controller1.domain.com' -and OSType -eq '
     Windows XP Service Pack 3'  }
```

EXAMPLE 2

This command returns all unconfigured machines which have registered with the site in the past, but which are not registered at present.

```
Get-BrokerUnconfiguredMachine -CurrentlyRegistered $false
```

EXAMPLE 3

This command returns all unconfigured machines which are currently registered with the site or which have registered with the site in the past.

```
Get-BrokerUnconfiguredMachine -Filter {
  CurrentlyRegistered -eq $true -or CurrentlyRegistered -eq $false  }
```

Parameters

-SID

Gets machines by their machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-MachineName

Gets machines by their machine name (in the form domain\machine).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AgentVersion

Gets machines with a specific Virtual Desktop Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ControllerDNSName

Gets machines by the DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-CurrentlyRegistered

Gets machines by whether they are currently registered with the site. If there is no CurrentlyRegistered filter, then the default is to return unconfigured machines that are currently registered.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets machines by their DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FunctionalLevel

Gets machines with a specific FunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LastDeregistrationTime

Gets machines by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets machines by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OSVersion

Gets machines by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-SessionSupport

Gets machines that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.UnconfiguredMachine

Get-BrokerUnconfiguredMachine returns an object for each matching machine

Notes

It is generally better to compare dates and times using -Filter and relative comparisons. See about_Broker_Filtering and the examples in this topic for more information.

Related Links

- about_Broker_Machines
- about_Broker_Filtering
- Get-BrokerMachine

Get-BrokerUniversalClaim

March 11, 2024

Get a broker Universal Claim

Syntax

```
Get-BrokerUniversalClaim
   [-VirtualSid <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerUniversalClaim
   -Claim <String>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Reads a specific UniversalClaim by Claim or VirtualSid, or enumerates UniversalClaims by passing nothing.

———————BrokerUniversalClaim Object

A UniversalClaim object stores the mapping between a Claim and a VirtualSID. A claim can be an arbitrary string up to 450 characters in length. Each Claim is mapped to a Virtual SID. Virtual SIDs are generated by either taking a SHA256 hash of the claim, and storing the hash bytes in a SID format, or in the case of a SID string being used as a Claim, the SID is copied to the VirtualSid.

- Claim (System.String)

  The Claim for this UniversalClaim mapping

- DirectoryContext (System.String)

  The DirectoryContext associated with this UniversalClaim mapping

- UniversalClaimsTenantContext (System.String)

  The UniversalClaimsTenantContext associated with this UniversalClaim mapping

- VirtualSid (System.String)

  The VirtualSid for this UniversalClaim mapping

Examples

EXAMPLE 1

Gets all broker UniversalClaim mappings

```
Get-BrokerUniversalClaim
```

Parameters

-Claim

The Claim for this UniversalClaim mapping

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-VirtualSid

The VirtualSid for this UniversalClaim mapping.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the

output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.UniversalClaim

Get-BrokerUniversalClaim returns an object for each matching broker UniversalClaim mapping.

Related Links

Get-BrokerUser

March 11, 2024

Gets broker users configured for this site.

Syntax

```
Get-BrokerUser
   [[-Name] <String>]
   [-DirectoryContext <String>]
   [-FullName <String>]
   [-HomeZoneName <String>]
   [-HomeZoneUid <Guid>]
   [-IdentityClaims <String>]
   [-NameLookupFailureCount <Int32>]
   [-PrimaryClaim <String>]
   [-SAMName <String>]
   [-UPN <String>]
   [-ApplicationGroupUid <Int32>]
   [-ApplicationUid <Int32>]
   [-SessionLingerDesktopGroupUid <Int32>]
   [-SessionPreLaunchDesktopGroupUid <Int32>]
   [-MachineUid <Int32>]
   [-PrivateDesktopUid <Int32>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-BrokerUser
   -SID <String>
   [-Property <String[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Retrieve broker users matching the specified criteria. If no parameters are specified this cmdlet enumerates all broker users.

For information about advanced filtering options, see about_Broker_Filtering.

——————————BrokerUser Object

The BrokerUser object represents a single instance of an user. It contains the following properties:

- DirectoryContext (System.String)

  The directory context of a user

- FullName (System.String)

  The full name of a user

- HomeZoneName (System.String)

  Name of home zone associated with user

- HomeZoneUid (System.Guid?)

  Home zone associated with this user

- IdentityClaims (System.String)

  The identity claims of a user

- Name (System.String)

  The name of a user

- NameLookupFailureCount (System.Int32)

  Tracks the number of consecutive directory lookup failures for this account

- PrimaryClaim (System.String)

  The primary identity claim of a user

- SAMName (System.String)

  The SAMName of a user

- SID (System.String)

  The SID of a user

- UPN (System.String)

  The UPN of a user

Examples

EXAMPLE 1

Get all broker user objects with names matching the supplied pattern

```
Get-BrokerUser DOMAIN7\*
```

EXAMPLE 2

Get all broker user objects added to the specified private desktop

```
$pdt = Get-BrokerPrivateDesktop DOMAIN\MACHINENAME\n
             Get-BrokerUser -PrivateDesktopUid $pdt.Uid
```

Parameters

-SID

Gets the broker user with the specified SID property value.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-Name

Gets the broker user with the specified Name property.

Type:	String
Position:	2
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-DirectoryContext

Gets the broker user with the specified directory context property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FullName

Gets the broker user with the specified FullName property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HomeZoneName

Gets user/group accounts having a home zone preference matching the specified name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HomeZoneUid

Gets user/group accounts having a home zone preference matching the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityClaims

Gets the broker user with the specified identity claims property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-NameLookupFailureCount

Tracks the number of consecutive directory lookup failures for this account.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrimaryClaim

Gets the broker user with the specified primary claim property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SAMName

Gets the broker user with the specified SAMName property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UPN

Gets the broker user with the specified UPN property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ApplicationGroupUid

Gets broker users associated with the application group with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets broker users associated with the application with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionLingerDesktopGroupUid

Gets broker users associated with the desktop group session linger settings with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionPreLaunchDesktopGroupUid

Gets broker users associated with the desktop group session pre-launch settings with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets broker users associated with the broker machine with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrivateDesktopUid

Gets broker users associated with the private desktop with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.User

Get-BrokerUser returns an object for each matching broker user.

Related Links

- Add-BrokerUser
- Remove-BrokerUser

Get-BrokerUserZonePreference

March 11, 2024

Gets user/group accounts with zone preferences configured for this site

Syntax

```
Get-BrokerUserZonePreference
    [[-Name] <String>]
    [-DirectoryContext <String>]
    [-FullName <String>]
    [-HomeZoneName <String>]
    [-HomeZoneUid <Guid>]
    [-IdentityClaims <String>]
    [-PrimaryClaim <String>]
```



```
    [-SAMName <String>]
    [-UPN <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-BrokerUserZonePreference
   -SID <String>
    [-Property <String[]>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve user/group account details with a home zone preference specified and matching the specified criteria. If no parameters are specified this cmdlet enumerates all account details with associated home zones preferences.

For information about advanced filtering options, see about_Broker_Filtering.

————————BrokerUserZonePreference Object

The BrokerUserZonePreference object represents a user/group account having an associated home zone preference to be used when launching desktop or application sessions. It contains the following properties:

- DirectoryContext (System.String)

  The directory context of a user

- FullName (System.String)

  Full name of user/group account.

- HomeZoneName (System.String)

  Name of home zone preference associated with user/group account for this site.

- HomeZoneUid (System.Guid)

  Home zone preference associated with user/group account for this site.

- IdentityClaims (System.String)

  The identity claims of a user

- Name (System.String)

  SAM name of user group/account (domain\user).

- PrimaryClaim (System.String)

  The primary identity claim of a user

- SAMName (System.String)

  The SAMName of a user

- SID (System.String)

  SID of user/group account.

- UPN (System.String)

  UPN of user/group account (user@domain).

Examples

EXAMPLE 1

Get all user/group accounts with a home zone preference whose names match the supplied pattern

```
1  Get-BrokerUserZonePreference DOMAIN7\*
```

EXAMPLE 2

Get all user/group accounts with the specified zone as their home zone preference.

```
1  Get-BrokerUserZonePreference -HomeZoneUid 79AD8059-05B7-4BAA-9369-5
       C74EF52D3EB
```

Parameters

-SID

Gets user/group accounts with a home zone preference and the specified SID.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-Name

Gets user/group accounts with a home zone preference and the specified SAM name (domain\user).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-DirectoryContext

Gets the broker user with the specified directory context property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-FullName

Gets user/group accounts with a home zone preference and the specified full name.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HomeZoneName

Gets user/group accounts having a home zone preference with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HomeZoneUid

Gets user/group accounts having a home zone preference with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityClaims

Gets the broker user with the specified identity claims property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PrimaryClaim

Gets the broker user with the specified primary claim property value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SAMName

The SAMName of a user

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UPN

Gets user/group accounts with a home zone preference and the specified UPN (user@domain).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.UserZonePreference

Get-BrokerUserZonePreference returns an object for each user/group account with a home zone preference.

Related Links

- New-BrokerUserZonePreference
- Set-BrokerUserZonePreference
- Remove-BrokerUserZonePreference
- Get-BrokerUser

Group-BrokerDesktop

March 11, 2024

Groups and counts desktops with the same value for a specified property.

Syntax

```
Group-BrokerDesktop
      [[-MachineName] <String>]
      [-AgentVersion <String>]
      [-ApplicationInUse <String>]
```



```
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-AssociatedUserFullName <String>]
    [-AssociatedUserName <String>]
    [-AssociatedUserUPN <String>]
    [-AutonomouslyBrokered <Boolean>]
    [-CatalogName <String>]
    [-CatalogUid <Int32>]
    [-ClientAddress <String>]
    [-ClientName <String>]
    [-ClientVersion <String>]
    [-ColorDepth <ColorDepth>]
    [-ConnectedViaHostName <String>]
    [-ConnectedViaIP <String>]
    [-ControllerDNSName <String>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DesktopCondition <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-DesktopKind <DesktopKind>]
    [-DeviceId <String>]
    [-DNSName <String>]
    [-FunctionalLevel <FunctionalLevel>]
    [-HardwareId <String>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-IconUid <Int32>]
    [-ImageOutOfDate <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAssigned <Boolean>]
    [-IsPhysical <Boolean>]
    [-LastConnectionFailure <ConnectionFailureReason>]
    [-LastConnectionTime <DateTime>]
    [-LastConnectionUser <String>]
    [-LastDeregistrationReason <DeregistrationReason>]
    [-LastDeregistrationTime <DateTime>]
    [-LastErrorReason <String>]
    [-LastErrorTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-LaunchedViaHostName <String>]
    [-LaunchedViaIP <String>]
    [-MachineInternalState <MachineInternalState>]
    [-MachineUid <Int32>]
    [-OSType <String>]
    [-OSVersion <String>]
    [-PersistUserChanges <PersistUserChanges>]
    [-PowerActionPending <Boolean>]
    [-PowerState <PowerState>]
```



```
      [-Protocol <String>]
      [-ProvisioningType <ProvisioningType>]
      [-PublishedApplication <String>]
      [-PublishedName <String>]
      [-PvdStage <PvdStage>]
      [-RegistrationState <RegistrationState>]
      [-SecureIcaActive <Boolean>]
      [-SecureIcaRequired <Boolean>]
      [-SessionHidden <Boolean>]
      [-SessionId <Int32>]
      [-SessionState <SessionState>]
      [-SessionStateChangeTime <DateTime>]
      [-SessionUid <Int64>]
      [-SessionUserName <String>]
      [-SessionUserSID <String>]
      [-SID <String>]
      [-SmartAccessTag <String>]
      [-StartTime <DateTime>]
      [-SummaryState <DesktopSummaryState>]
      [-Tag <String>]
      [-WillShutdownAfterUse <Boolean>]
      [-ApplicationUid <Int32>]
      -Property <String>
      [-ReturnTotalRecordCount]
      [-MaxRecordCount <Int32>]
      [-Skip <Int32>]
      [-SortBy <String>]
      [-Filter <String>]
      [-FilterScope <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Group-BrokerDesktop
      [-Uid] <Int32>
      -Property <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet is now deprecated, please use Group-BrokerMachine.

Filters desktops using the specified criteria, then groups and counts matching desktops with the same value for a particular property. The number of desktops in the group, and the property value for the group, is output. For example:

```
Group-BrokerDesktop -Property SummaryState
Count Name
----- ----
   43 Available
   17 InUse
```



```
6       3 Disconnected
```

Filtering supports the same options as the Get-BrokerDesktop cmdlet, and allows filtering on both desktop and session properties.

Group-BrokerDesktop is similar to the standard PowerShell Group-Object, but is faster than piping the output of Get-BrokerDesktop into Group-Object when working with many desktops.

Note that all session information properties for multi-session desktops is always $null, this means that is is not possible to group these desktops by session information using this command. Use Get-BrokerSession to get information on all current sessions.

Also note that the MaxRecordCount, ReturnTotalRecordCount, Skip, and SortBy parameters apply to GroupInfo records output rather than the filtered desktops.

Examples

EXAMPLE 1

Group desktops from the dg1 group by summary state.

```
1 Group-BrokerDesktop -Property SummaryState -DesktopGroupName dg1
```

EXAMPLE 2

For desktops where the last connection attempt failed, list the most common reason for failure, ignoring connections that failed over a week ago.

```
1 Group-BrokerDesktop -Property LastConnectionFailure -Filter {
2   LastConnectionFailure -ne "None" -and LastConnectionTime -ge '-7'  }
3   -MaxRecordCount 1
```

EXAMPLE 3

List alphabetically the hypervisor servers hosting desktops that are currently experiencing high network latency.

```
1 Group-BrokerDesktop -Property HostingServerName -DesktopCondition
      ICALatency -SortBy Name
```


Parameters

-Uid

Gets desktops with a specific UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Selects the property by which matching desktops are grouped.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets desktops with a specific machine name (in the form ‘domain\machine’).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AgentVersion

Gets desktops with a specific Citrix Virtual Delivery Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationInUse

Gets desktops running a specified published application (identified by browser name).

String comparisons are case-insensitive.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedClientName

Gets desktops assigned to a specific client name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedIPAddress

Gets desktops assigned to a specific IP address.q

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserFullName

Gets desktops with an associated user identified by their full name (usually in the form ‘first-name last-name’).

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserName

Gets desktops with an associated user identified by their user name (in the form ‘domain\user’).

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserUPN

Gets desktops with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are the current user for shared desktops, and the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutonomouslyBrokered

Gets desktops according to whether their current session is autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets desktops from the catalog with the specific name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUid

Gets desktops from a catalog with a specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientAddress

Gets desktops with a specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientName

Gets desktops with a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientVersion

Gets desktops with a specific client version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Gets desktops configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectedViaHostName

Gets desktops with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectedViaIP

Gets desktops with a specific IP address of the incoming connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets desktops with a specific DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeliveryType

Gets desktops of a particular delivery type.

Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Get desktops with a specific description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopCondition

Gets desktop with an outstanding desktop condition condition.

Valid values are:

- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets desktops from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets desktops from a desktop group with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Deprecated: Use AllocationType parameter.

Gets desktops of a particular kind.

Valid values are Private, Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeviceId

Gets desktops with a specific client device ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Get desktops with a specific DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FunctionalLevel

Gets desktops with a specific FunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HardwareId

Gets desktops with a specific client hardware ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineId

Gets desktops with a specific machine ID known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineName

Gets desktops with a specific machine name known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingServerName

Gets desktops with a specific name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionName

Gets desktops with a specific name of the hosting hypervisor connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionUid

Gets desktops with a specific UID of the hosting hypervisor connection.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets desktops with a specific configured icon. Note that desktops with a null IconUid use the icon of the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets desktops if they have an ImageOutOfDate flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets desktops with a specific InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets desktops with a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsAssigned

Gets desktops according to whether they are assigned or not. Desktops may be assigned to one or more users or groups, a client IP address or a client endpoint name.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPhysical

Specifies if machines in the catalog can be power managed by the Citrix Broker Service. Where the power state of the machine cannot be controlled, specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the deprecated CatalogKind parameter only with a Pvs catalog kind.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionFailure

Gets desktops with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the desktop yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

Type:	ConnectionFailureReason

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionTime

Gets desktops that last connected at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionUser

Gets desktops where a specific user name last attempted a connection (in the form 'domain\user').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationReason

Gets desktops whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationTime

Gets desktops that were last deregistered by a specific time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorReason

Gets desktops with the specified last error reason.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorTime

Gets desktops with the specified last error time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets desktops with a specific time that the hosting information was last updated.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaHostName

Gets desktops with a specific host name of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaIP

Gets desktops with a specific IP address of the StoreFront server from which the user launched the session.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineInternalState

Gets desktops with the specified internal machine state.

Type:	MachineInternalState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets desktops with a specific machine UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets desktops by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSVersion

Gets desktops by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PersistUserChanges

Gets desktops by the location where the user changes are persisted.

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerActionPending

Gets desktops with a specific power action pending state.

Valid values are $true or $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets desktops with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Protocol

Gets desktops with connections using a specific protocol, for example HDX, RDP, or Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningType

Specifies the provisioning type for the catalog. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,…).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedApplication

Gets desktops with a specific application published on them (identified by its browser name).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Gets desktops with a specific published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdStage

This property is no longer supported.

Type:	PvdStage
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RegistrationState

Gets desktops with a specific registration state.

Valid values are Unregistered, Initializing, Registered and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaActive

Gets desktops depending on whether the current session uses SecureICA or not.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Gets desktops configured with a particular SecureIcaRequired setting. Note that the desktop setting of $null indicates that the desktop group value is used.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionHidden

Gets desktops by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Session properties are always null for multi-session desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionId

Deprecated.

Gets desktops by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Gets desktops with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, Non-BrokeredSession, and Unknown.

Session properties are always null for multi-session desktops.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Gets desktops whose sessions last changed state at a specific time.

Session properties are always null for multi-session desktops.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets desktops with a specific session UID ($null for no session).

Session properties are always null for multi-session desktops.

Type:	Int64
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserName

Gets desktops with a specific user name for the current session (in the form ‘domain\user’).

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserSID

Gets desktops with a specific SID of the current session user.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Gets desktops with a specific machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SmartAccessTag

Gets desktops where the session has the specific SmartAccess tag.

Session properties are always null for multi-session desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Gets desktops with a specific session start time.

Session properties are always null for multi-session desktops.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SummaryState

Gets desktops with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, and InUse.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets desktops with a specific tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WillShutdownAfterUse

Gets desktops depending on whether they shut down after use or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets desktops with a specific published application (identified by its UID).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GroupInfo

Each GroupInfo object represents one group, and contains the following properties:

- Count: The count of desktops in this group.
- Name: The value of the property the desktops were grouped by (as a string).

If you do not specify -SortBy, groups are sorted with the largest count first.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Filtering
- about_Broker_Desktops
- Get-BrokerDesktop
- Group-Object

Group-BrokerMachine

March 11, 2024

Groups and counts machines with the same value for a specified property.

Syntax

```
Group-BrokerMachine
      [[-MachineName] <String>]
      [-AgentVersion <String>]
      [-AllocationType <AllocationType>]
      [-ApplicationInUse <String>]
      [-AssignedClientName <String>]
      [-AssignedIPAddress <String>]
      [-AssignedUserSID <String>]
      [-AssociatedTenantId <Guid>]
      [-AssociatedUserFullName <String>]
```



```
    [-AssociatedUserName <String>]
    [-AssociatedUserSID <String>]
    [-AssociatedUserUPN <String>]
    [-AzureADJoinedMode <String>]
    [-AzureDeviceId <String>]
    [-BrowserName <String>]
    [-CatalogName <String>]
    [-CatalogUid <Int32>]
    [-CatalogUUID <Guid>]
    [-CbpVersion <CBPVersion>]
    [-ColorDepth <ColorDepth>]
    [-ControllerDNSName <String>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DesktopCondition <String>]
    [-DesktopGroupName <String>]
    [-DesktopGroupUid <Int32>]
    [-DesktopGroupUUID <Guid>]
    [-DesktopKind <DesktopKind>]
    [-DesktopUid <Int32>]
    [-DNSName <String>]
    [-DrainingUntilShutdown <Boolean>]
    [-FaultState <MachineFaultState>]
    [-FunctionalLevel <FunctionalLevel>]
    [-HostedMachineId <String>]
    [-HostedMachineName <String>]
    [-HostingServerName <String>]
    [-HypervisorConnectionName <String>]
    [-HypervisorConnectionUid <Int32>]
    [-HypHypervisorConnectionUid <Guid>]
    [-IconUid <Int32>]
    [-ImageOutOfDate <Boolean>]
    [-InMaintenanceMode <Boolean>]
    [-IPAddress <String>]
    [-IsAssigned <Boolean>]
    [-IsPhysical <Boolean>]
    [-IsReserved <Boolean>]
    [-LastAssignmentTime <DateTime>]
    [-LastConnectionFailure <ConnectionFailureReason>]
    [-LastConnectionTime <DateTime>]
    [-LastConnectionUser <String>]
    [-LastDeregistrationReason <DeregistrationReason>]
    [-LastDeregistrationTime <DateTime>]
    [-LastErrorReason <String>]
    [-LastErrorTime <DateTime>]
    [-LastHostingUpdateTime <DateTime>]
    [-LastPvdErrorReason <String>]
    [-LastPvdErrorTime <DateTime>]
    [-LastRegistrationTime <DateTime>]
    [-LoadIndex <Int32>]
    [-MacAddress <String>]
    [-MachineInternalState <MachineInternalState>]
    [-MachineUnavailableReason <String>]
```



```
64        [-MaintenanceModeReason <MaintenanceModeReason>]
65        [-Metadata <String>]
66        [-NameLookupFailureCount <Int32>]
67        [-OSType <String>]
68        [-OSVersion <String>]
69        [-PersistUserChanges <PersistUserChanges>]
70        [-PowerActionPending <Boolean>]
71        [-PowerState <PowerState>]
72        [-ProvisioningType <ProvisioningType>]
73        [-PublishedApplication <String>]
74        [-PublishedName <String>]
75        [-PvdEstimatedCompletionTime <DateTime>]
76        [-PvdPercentDone <Int32>]
77        [-PvdStage <PvdStage>]
78        [-PvdUpdateStartTime <DateTime>]
79        [-RegistrationState <RegistrationState>]
80        [-ScheduledReboot <ScheduledReboot>]
81        [-SecureIcaRequired <Boolean>]
82        [-SessionAutonomouslyBrokered <Boolean>]
83        [-SessionClientAddress <String>]
84        [-SessionClientName <String>]
85        [-SessionClientVersion <String>]
86        [-SessionConnectedViaHostName <String>]
87        [-SessionConnectedViaIP <String>]
88        [-SessionCount <Int32>]
89        [-SessionDeviceId <String>]
90        [-SessionHardwareId <String>]
91        [-SessionHidden <Boolean>]
92        [-SessionKey <Guid>]
93        [-SessionLaunchedViaHostName <String>]
94        [-SessionLaunchedViaIP <String>]
95        [-SessionProtocol <String>]
96        [-SessionSecureIcaActive <Boolean>]
97        [-SessionsEstablished <Int32>]
98        [-SessionSmartAccessTag <String>]
99        [-SessionsPending <Int32>]
100       [-SessionStartTime <DateTime>]
101       [-SessionState <SessionState>]
102       [-SessionStateChangeTime <DateTime>]
103       [-SessionSupport <SessionSupport>]
104       [-SessionType <SessionType>]
105       [-SessionUid <Int64>]
106       [-SessionUserName <String>]
107       [-SessionUserSID <String>]
108       [-SID <String>]
109       [-SummaryState <DesktopSummaryState>]
110       [-SupportedPowerAction <String>]
111       [-Tag <String>]
112       [-UUID <Guid>]
113       [-VMToolsState <VMToolsState>]
114       [-WillShutdownAfterUse <Boolean>]
115       [-WillShutdownAfterUseReason <WillShutdownAfterUseReason>]
116       [-WindowsConnectionSetting <WindowsConnectionSetting>]
```



```
       [-ZoneHealthy <Boolean>]
       [-ZoneName <String>]
       [-ZoneUid <Guid>]
       -Property <String>
       [-ReturnTotalRecordCount]
       [-MaxRecordCount <Int32>]
       [-Skip <Int32>]
       [-SortBy <String>]
       [-Filter <String>]
       [-FilterScope <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Group-BrokerMachine
      [-Uid] <Int32>
      -Property <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Filters machines using the specified criteria, then groups and counts matching machines with the same value for a particular property. The number of machines in the group, and the property value for the group, is output. For example:

C:\PS> Group-BrokerMachine -Property SummaryState

Count Name

---

43 Available

17 InUse

3 Disconnected

Filtering supports the same options as the Get-BrokerMachine cmdlet, and allows filtering on both machine and session properties.

Group-BrokerMachine is similar to the standard PowerShell Group-Object, but is faster than piping the output of Get-BrokerMachine into Group-Object when working with many machines.

Note that the MaxRecordCount, ReturnTotalRecordCount, Skip, and SortBy parameters apply to GroupInfo records output rather than the filtered machines.

Examples

EXAMPLE 1

Group machines from the dg1 group by summary state.

```
Group-BrokerMachine -Property SummaryState -DesktopGroupName dg1
```

EXAMPLE 2

For machines where the last connection attempt failed, list the most common reason for failure, ignoring connections that failed over a week ago.

```
Group-BrokerMachine -Property LastConnectionFailure -Filter {
  LastConnectionFailure -ne "None" -and LastConnectionTime -ge '-7'  }
  -MaxRecordCount 1
```

EXAMPLE 3

List alphabetically the hypervisor servers hosting machines that are currently experiencing high network latency.

```
Group-BrokerMachine -Property HostingServerName -DesktopCondition
    ICALatency -SortBy Name
```

Parameters

-Uid

Gets a machine with a specific UID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Selects the property by which matching machines are grouped.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets machines with a specific machine name (in the form domain\machine).

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AgentVersion

Gets machines with a specific Virtual Delivery Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllocationType

Gets machines from catalogs with the specified allocation type.

Type:	AllocationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationInUse

Gets machines running a specified published application. String comparisons are case-insensitive.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedClientName

Gets machines that have been assigned to the specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedIPAddress

Gets machines that have been assigned to the specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedUserSID

Gets machines with the specific SID of the user to whom the desktop is assigned.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedTenantId

Gets machines associated with the specified tenant.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserFullName

Gets machines with an associated user identified by their full name (usually 'first-name last-name').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserName

Gets machines with an associated user identified by their user name (in the form 'domain\user').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserSID

Gets machines with an associated user identified by their Windows SID.

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssociatedUserUPN

Gets machines with an associated user identified by their User Principle Name (in the form 'user@domain').

Associated users are all current users of a desktop, plus the assigned users for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureADJoinedMode

Gets machines with a specific Azure AD Domain Join Type

- NotAadJoined - Machine not joined to Azure AD yet.
- HybridAadJoined - Machine was Hybrid Aad joined.
- PureAadJoined - Machine was Pure Aad joined.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDeviceId

Gets machines with matching Azure DeviceId.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrowserName

Gets assigned machines backing desktop resources that have browser names matching the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets machines from the catalog with the specific name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUid

Gets machines from the catalog with the specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogUUID

Gets machines from the catalog with the specific UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CbpVersion

The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

Type:	CBPVersion
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Gets machines configured with a specific color depth.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets machines by the DNS name of the controller they are registered with.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeliveryType

Gets machines of a particular delivery type.

Valid values are AppsOnly, DesktopsOnly, DesktopsAndApps

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Description

Get machines by description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopCondition

Gets machines with an outstanding desktop condition.

Valid values are:

- CPU: Indicates the machine has high CPU usage
- ICALatency: Indicates the network latency is high
- UPMLogonTime: Indicates that the profile load time was high

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets machines from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUid

Gets machines from a desktop group with a specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupUUID

Gets machines from a desktop group with a specific UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Deprecated: Use AllocationType parameter.

Gets machines of a particular kind.

Valid values are Private, Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopUid

Gets the machine that corresponds to the desktop with the specific UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets machines with the specific DNS name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DrainingUntilShutdown

Gets machines depending on whether they are draining until shutdown or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FaultState

Gets machines currently in the specified fault state.

Type:	MachineFaultState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FunctionalLevel

Gets machines with a specific FunctionalLevel.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineId

Gets machines with the specific machine ID known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineName

Gets machines with the specific machine name known to the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingServerName

Gets machines by the name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionName

Gets machines with the specific name of the hypervisor connection hosting them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionUid

Gets machines with the specific UID of the hypervisor connection hosting them.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypHypervisorConnectionUid

Gets machines with the specific UUID of the hypervisor connection hosting them.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Gets machines by configured icon. Note that machines with a null IconUid use the icon of the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets machines depending on whether their disk image is out of date or not (for machines provisioned using MCS only).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets machines by whether they are in maintenance mode or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-IPAddress

Gets machines with a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsAssigned

Gets machines according to whether they are assigned or not. Machines may be assigned to one or more users or groups, a client IP address or a client endpoint name.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPhysical

Gets machines according to whether they can be power managed by XenDesktop or not.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsReserved

Gets machines that are reserved for special use, for example, for AppDisk preparation.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastAssignmentTime

Gets machines with the specific LastAssignmentTime.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionFailure

Gets machines with a specific reason for the last recorded connection failure. This value is None if the last connection was successful or if there has been no attempt to connect to the machine yet.

Valid values are None, SessionPreparation, RegistrationTimeout, ConnectionTimeout, Licensing, Ticketing, and Other.

Type:	ConnectionFailureReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionTime

Gets machines to which a user session connection occurred at a specific time. This is the time that the broker detected that the connection attempt either succeeded or failed.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastConnectionUser

Gets machines where a specific user name last attempted a connection (in the form 'domain\user').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationReason

Gets machines whose broker last recorded a specific deregistration reason.

Valid values are $null, AgentShutdown, AgentSuspended, AgentRequested, IncompatibleVersion, AgentAddressResolutionFailed, AgentNotContactable, AgentWrongActiveDirectoryOU, EmptyRegistrationRequest, MissingRegistrationCapabilities, MissingAgentVersion, InconsistentRegistrationCapabilities, NotLicensedForFeature, UnsupportedCredentialSecurityVersion, InvalidRegistrationRequest, SingleMultiSessionMismatch, FunctionalLevelTooLowForCatalog, FunctionalLevelTooLowForDesktopGroup, PowerOff, DesktopRestart, DesktopRemoved, AgentRejectedSettingsUpdate, SendSettingsFailure, SessionAuditFailure, SessionPrepareFailure, ContactLost, SettingsCreationFailure, UnknownError and BrokerRegistrationLimitReached.

Type:	DeregistrationReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastDeregistrationTime

Gets machines by the time that they were last deregistered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorReason

Gets machines with the specified last error reason.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastErrorTime

Gets machines with the specified last error time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastHostingUpdateTime

Gets machines with a specific time that the hosting information was last updated or the completion of the last power action.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastPvdErrorReason

This property is no longer supported.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastPvdErrorTime

This property is no longer supported.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastRegistrationTime

Gets machines by the time that they last registered.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoadIndex

Gets machines by their current load index.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MacAddress

Gets machines with a specific MAC address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineInternalState

Gets machines with the specified internal state.

Type:	MachineInternalState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUnavailableReason

Gets machines that corresponds to a particular MachineUnavailable Reason

- None - No detailed reason specified
- LoadManagementInitializing - VDA load management logic currently initialising. Only occurs for multi-session capable VDAs.
- GctConnectionInitializing - VDA is still initializing control connection with NGS.
- AzureADJoinInitializing - VDA is still initializing Aad domain join.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaintenanceModeReason

Gets machines by the maintenance mode reason. Valid values are:

- None - Machine is not in maintenance mode.
- Administrator - Machine was manually placed in maintenance mode by an administrator.
- MaxFailedRegistrations - Machine was automatically placed in maintenance mode due to reaching the maximum failed registration limit.

Type:	MaintenanceModeReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NameLookupFailureCount

Tracks the number of consecutive directory lookup failures for this account.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OSType

Gets machines by the type of operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-OSVersion

Gets machines by the version of the operating system they are running.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PersistUserChanges

Gets machines according to the location where user changes are persisted. Values can be:

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerActionPending

Gets machines depending on whether a power action is pending or not.

Valid values are $true or $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets machines with a specific power state.

Valid values are Unmanaged, Unknown, Unavailable, Off, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningType

Specifies the provisioning type for the catalog. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedApplication

Gets machines with a specific application published to them (identified by its browser name).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Gets desktops with a specific published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdEstimatedCompletionTime

This property is no longer supported.

Type:	DateTime
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdPercentDone

This property is no longer supported.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdStage

This property is no longer supported.

Type:	PvdStage
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvdUpdateStartTime

This property is no longer supported.

Type:	DateTime
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RegistrationState

Gets machines in a specific registration state.

Valid values are Unregistered, Initializing, Registered, and AgentError.

Type:	RegistrationState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScheduledReboot

Gets machines according to their current status with respect to any scheduled reboots (for either scheduled desktop group reboots or image rollout purposes). Valid values are:

- None - No reboot currently scheduled.
- Pending - Reboot scheduled but machine still available for use.
- Draining - Reboot scheduled. New logons are disabled, but reconnections to existing sessions are allowed.
- InProgress - Machine is actively being rebooted.
- Natural - Natural reboot in progress. Machine is awaiting a restart.

Type:	ScheduledReboot
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Gets machines configured with a particular SecureIcaRequired setting. Note that the machine setting of $null indicates that the desktop group value is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionAutonomouslyBrokered

Gets machines according to whether their current session is autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Session properties are always null for multi-session machines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionClientAddress

Gets machines with a specific client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionClientName

Gets machines with a specific client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionClientVersion

Gets machines with a specific client version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionConnectedViaHostName

Gets machines with a specific host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionConnectedViaIP

Gets machines with a specific IP address of the incoming connection.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionCount

Gets machines according to the total number of both pending and established user sessions on the machine.

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionDeviceId

Gets machines with a specific client device ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionHardwareId

Gets machines with a specific client hardware ID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionHidden

Gets machines by whether their sessions are hidden or not. Hidden sessions are treated as though they do not exist when launching sessions using XenDesktop; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionKey

Gets machine running the session with the specified unique key.

Session properties are always null for multi-session machines.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionLaunchedViaHostName

Gets machines with a specific host name of the Web Interface server from which the user launched the session.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionLaunchedViaIP

Gets machines with a specific IP address of the Web Interface server from which the user launched the session.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionProtocol

Gets machines with connections using a specific protocol, for example HDX, RDP, or Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSecureIcaActive

Gets machines depending on whether the current session uses SecureICA or not.

Session properties are always null for multi-session machines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SessionsEstablished

Gets machines according to the number of established user sessions present on the machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSmartAccessTag

Gets session machines where the session has the specific SmartAccess tag.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionsPending

Get machines according to the number of pending user sessions for the machine.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStartTime

Gets machines with a specific session start time.

Session properties are always null for multi-session machines.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Gets machines with a specific session state.

Valid values are $null, Other, PreparingSession, Connected, Active, Disconnected, Reconnecting, Non-BrokeredSession, and Unknown.

Session properties are always null for multi-session machines.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Gets machines whose sessions last changed state at a specific time.

Session properties are always null for multi-session machines.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSupport

Gets machines that have the specified session capability. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionType

Gets machines with a specific session state.

Session properties are always null for multi-session machines.

Type:	SessionType
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUid

Gets single-session machines with a specific session UID ($null for no session).

Session properties are always null for multi-session machines.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserName

Gets machines with a specific user name for the current session (in the form 'domain\user').

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionUserSID

Gets machines with a specific SID of the current session user.

Session properties are always null for multi-session machines.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SID

Gets machines with a specific machine SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SummaryState

Gets machines with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, and InUse.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SupportedPowerAction

Gets machines that support the specified power action.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

Gets machines where the session has the given SmartAccess tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UUID

Gets machines with the specified value of UUID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMToolsState

Gets machines with a specific VM tools state.

Valid values are NotPresent, Unknown, NotStarted, and Running.

Type:	VMToolsState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WillShutdownAfterUse

Gets machines depending on whether they shut down after use or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WillShutdownAfterUseReason

Gets machines by the will shutdown after use reason. Valid values are:

- None - Machine will not shutdown after use.
- ResetDiskImage - Machine will shutdown after use to reset its disk image.
- ScheduledNaturalReboot - Machine will shutdown after use as part of the scheduled natural reboot process.
- OnDemandNaturalReboot - Machine will shutdown after use as part of an on-demand natural reboot process.

Type:	WillShutdownAfterUseReason
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WindowsConnectionSetting

Gets machines according to their current Windows connection setting (logon mode). Valid values are:

- LogonEnabled - All logons are enabled.
- Draining - New logons are disabled, but reconnections to existing sessions are allowed.
- DrainingUntilRestart - Same as Draining, but setting reverts to LogonEnabled when machine next restarts.
- LogonDisabled - All logons and reconnections are disabled.

This is a Windows setting and is not controlled by XenDesktop. It applies only to multi-session machines; for single-session machines its value is always LogonEnabled.

Type:	WindowsConnectionSetting
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneHealthy

Gets machines located in the zone with the specified health state.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets machines located in the zone with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Gets machines located in the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GroupInfo

Each GroupInfo object represents one group, and contains the following properties:

- Count: The count of machines in this group.
- Name: The value of the property the machines were grouped by (as a string).

If you do not specify -SortBy, groups are sorted with the largest count first.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Filtering
- about_Broker_Machines
- Get-BrokerMachine
- Group-Object

Group-BrokerSession

March 11, 2024

Groups and counts sessions with the same value for a specified property.

Syntax

```
Group-BrokerSession
      [[-SessionKey] <Guid>]
      [-AgentVersion <String>]
      [-AnonymousUserId <String>]
      [-ApplicationInUse <String>]
      [-AppState <SessionAppState>]
      [-AppStateLastChangeTime <DateTime>]
      [-AutonomouslyBrokered <Boolean>]
      [-BrokeringDuration <Int32>]
      [-BrokeringTime <DateTime>]
      [-BrokeringUserName <String>]
      [-BrokeringUserSID <String>]
      [-CatalogName <String>]
      [-CbpVersion <CBPVersion>]
      [-ClientAddress <String>]
      [-ClientName <String>]
      [-ClientPlatform <String>]
      [-ClientProductId <Int32>]
      [-ClientVersion <String>]
      [-ConnectedViaHostName <String>]
      [-ConnectedViaIP <String>]
      [-ConnectionMode <ConnectionMode>]
      [-ControllerDNSName <String>]
      [-DesktopGroupName <String>]
```



```
25        [-DesktopGroupUid <Int32>]
26        [-DesktopKind <DesktopKind>]
27        [-DesktopSID <String>]
28        [-DesktopUid <Int32>]
29        [-DeviceId <String>]
30        [-DNSName <String>]
31        [-EntitlementPolicyRuleUid <Int32>]
32        [-EstablishmentDuration <Int32>]
33        [-EstablishmentTime <DateTime>]
34        [-HardwareId <String>]
35        [-Hidden <Boolean>]
36        [-HostedMachineName <String>]
37        [-HostingServerName <String>]
38        [-HypervisorConnectionName <String>]
39        [-IdleDuration <TimeSpan>]
40        [-IdleSince <DateTime>]
41        [-ImageOutOfDate <Boolean>]
42        [-InMaintenanceMode <Boolean>]
43        [-IPAddress <String>]
44        [-IsAnonymousUser <Boolean>]
45        [-IsPhysical <Boolean>]
46        [-LaunchedViaHostName <String>]
47        [-LaunchedViaIP <String>]
48        [-LaunchedViaPublishedName <String>]
49        [-LaunchedViaWorkspace <Boolean>]
50        [-LogoffInProgress <Boolean>]
51        [-LogonInProgress <Boolean>]
52        [-MachineName <String>]
53        [-MachineSummaryState <DesktopSummaryState>]
54        [-MachineUid <Int32>]
55        [-Metadata <String>]
56        [-OSType <String>]
57        [-PersistUserChanges <PersistUserChanges>]
58        [-PowerState <PowerState>]
59        [-PreferredZoneName <String>]
60        [-PreferredZoneUid <Guid>]
61        [-Protocol <String>]
62        [-ProvisioningType <ProvisioningType>]
63        [-ReceiverIPAddress <String>]
64        [-ReceiverName <String>]
65        [-SecureIcaActive <Boolean>]
66        [-SessionId <Int32>]
67        [-SessionReconnection <SessionReconnection>]
68        [-SessionState <SessionState>]
69        [-SessionStateChangeTime <DateTime>]
70        [-SessionSupport <SessionSupport>]
71        [-SessionType <SessionType>]
72        [-StartTime <DateTime>]
73        [-TenantId <Guid>]
74        [-UntrustedUserName <String>]
75        [-UserFullName <String>]
76        [-UserName <String>]
77        [-UserSID <String>]
```



```
      [-UserUPN <String>]
      [-ZoneName <String>]
      [-ZoneUid <Guid>]
      [-ApplicationUid <Int32>]
      [-SharedDesktopUid <Int32>]
      -Property <String>
      [-ReturnTotalRecordCount]
      [-MaxRecordCount <Int32>]
      [-Skip <Int32>]
      [-SortBy <String>]
      [-Filter <String>]
      [-FilterScope <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Group-BrokerSession
      [-Uid] <Int64>
      -Property <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Filters sessions using the specified criteria, then groups and counts matching sessions with the same value for a particular property. The number of sessions in the group, and the property value for the group, is output. For example:

C:\PS> Group-BrokerSession -Property SessionState

Count Name

---

43 Active

17 NonBrokeredSession

3 Disconnected

Filtering supports the same options as the Get-BrokerSession cmdlet, and allows filtering on both machine and session properties.

Group-BrokerSession is similar to the standard PowerShell Group-Object, but is faster than piping the output of Get-BrokerSession into Group-Object when working with many machines.

Note that the MaxRecordCount, ReturnTotalRecordCount, Skip, and SortBy parameters apply to GroupInfo records output rather than the filtered sessions.

Examples

EXAMPLE 1

Group sessions on machines from the dg1 group by session state.

```
Group-BrokerSession -Property SessionState -DesktopGroupName dg1
```

EXAMPLE 2

List alphabetically the names of the clients connected to the site, but only show clients whose names starts with 'ThinClient'.

```
Group-BrokerSession -Property ClientName -ClientName 'ThinClient*' -
    SortBy Name
```

Parameters

-Uid

Gets session by its Uid.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Selects the property by which matching sessions are grouped.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-SessionKey

Gets session having the specified unique key.

Type:	Guid
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AgentVersion

Gets sessions with a specific Virtual Desktop Agent version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AnonymousUserId

Gets sessions associated with the specified user ID.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationInUse

Gets sessions running specific applications (identified by their SDK Name property).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppState

Gets sessions by their app state.

Valid values are PreLogon, PreLaunched, Active, Desktop, Lingering and NoApps.

Type:	SessionAppState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppStateLastChangeTime

Get sessions by their app state change time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutonomouslyBrokered

Gets sessions according to whether they are autonomously brokered or not. Autonomously brokered sessions are HDX sessions established by direct connection without being brokered.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringDuration

Gets session with a specific time taken to broker. In general, Citrix recommends using -Filter and relative comparisons.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringTime

Get sessions brokered at a specific time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringUserName

Get sessions by brokering user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrokeringUserSID

Get sessions by brokering user SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CatalogName

Gets sessions on machines from a specific catalog name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CbpVersion

The version of CBP that the VDA is currently registered with. This will be null when the VDA is not registered.

Type:	CBPVersion
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientAddress

Get sessions by client IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientName

Get sessions by client name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientPlatform

Get sessions by client platform.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientProductId

Get sessions by client product ID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientVersion

Get sessions by client version.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectedViaHostName

Get sessions by host name of the incoming connection. This is usually a proxy or Citrix Access Gateway server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectedViaIP

Get sessions by IP address of the incoming connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectionMode

Gets sessions by the way in which the most recent connection to the session was established.

Valid modes are Brokered, Unbrokered, LeasedConnection, VdaHighAvailabilityMode, ThirdPartyBroker, and ThirdPartyBrokerWithLicensing.

Type:	ConnectionMode
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerDNSName

Gets sessions that are hosted on machines which are registered with a specific controller.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupName

Gets sessions from a desktop group with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-DesktopGroupUid

Gets sessions from a desktop group with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopKind

Gets sessions on a desktop of a particular kind.

Valid values are Private and Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopSID

Get sessions by desktop SID.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopUid

Get sessions by desktop Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeviceId

Get sessions by client device id.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DNSName

Gets sessions by their machine's DNS name.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EntitlementPolicyRuleUid

Gets sessions where the user was granted the entitlement to launch the session from the specified entitlement policy rule.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EstablishmentDuration

Gets sessions which took a specific time to establish. In general, Citrix recommends using -Filter and relative comparisons.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EstablishmentTime

Gets sessions which became established at a particular time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HardwareId

Get sessions by client hardware id.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Hidden

Get sessions by whether they are hidden or not. Hidden sessions are treated as though they do not exist when brokering sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineName

Gets sessions by their machine's name as known to its hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingServerName

Gets sessions hosted by a machine with a specific name of the hosting hypervisor server.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionName

Gets sessions hosted by a machine with a specific name of the hosting hypervisor connection.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdleDuration

Gets sessions that have been idle for the specified period

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IdleSince

Time at which session went idle

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ImageOutOfDate

Gets sessions hosted by a machine with a specific ImageOutOfDate setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Gets sessions hosted by a machine with a specific InMaintenanceMode setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IPAddress

Gets sessions hosted by a machine with a specific IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsAnonymousUser

Gets sessions depending on whether they were established anonymously ($true) or not ($false). An anonymous session is established without user credentials and a temporary local user account is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-IsPhysical

Gets sessions hosted on machines where the flag indicating if the machine can be power managed by the Citrix Broker Service matches the requested value. Where the power state of the machine cannot be controlled, specify $true, otherwise $false.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaHostName

Get sessions by the host name of the StoreFront server from which a user launches a session.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaIP

Get sessions by the IP address of the StoreFront server from which a user launches a session.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaPublishedName

Gets sessions originally launched using a resource having a name matching the specified published name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LaunchedViaWorkspace

Gets sessions by whether they were launched via Citrix Workspace or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffInProgress

Gets sessions by whether they are in the process of being logged off or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogonInProgress

Gets sessions by whether they are still executing user logon processing or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineName

Gets sessions by their machine name (in the form DOMAIN\machine).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineSummaryState

Gets sessions on a machine with a specific summary state.

Valid values are Off, Unregistered, Available, Disconnected, Preparing, and InUse.

Type:	DesktopSummaryState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineUid

Gets sessions on a machine with the specified UID.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-OSType

Gets sessions with a specific type of operating system.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PersistUserChanges

Gets sessions where the user changes are persisted in a particular manner. Values can be:

- OnLocal - User changes are persisted locally.
- Discard - User changes are discarded.

Type:	PersistUserChanges
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerState

Gets sessions on machines in the specified power state.

Valid values are Unmanaged, Unknown, Unavailable, On, Suspended, TurningOn, TurningOff, Suspending, and Resuming.

Type:	PowerState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredZoneName

Gets sessions originally launched with the specified preferred zone name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredZoneUid

Gets sessions originally launched with the specified preferred zone Uid.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Protocol

Get sessions by connection protocol. Valid values are HDX, RDP and Console.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningType

Gets sessions hosted on machines provisioned in a particular manner. Values can be:

- Manual - No automated provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,...).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReceiverIPAddress

Gets sessions with the specified client IP address supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ReceiverName

Gets sessions with the specified client name supplied by Receiver (for example, StoreFront) when the session was launched, or reconnected.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaActive

Get sessions by their use of SecureICA.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionId

Deprecated.

Gets sessions by session ID, a unique identifier that Remote Desktop Services uses to track the session but it is only unique on that machine.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Get sessions by their session reconnection (roaming) behavior. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionState

Get sessions by their state.

Valid values are Other, PreparingNewSession, Connected, Active, Disconnected, Reconnecting, Non-BrokeredSession, and Unknown.

Type:	SessionState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionStateChangeTime

Get sessions by their last state change time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSupport

Gets sessions hosted on machines which support the required pattern of sessions. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionType

Gets sessions by their type.

Valid values are Application and Desktop.

Type:	SessionType
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Gets sessions by their start time. In general, Citrix recommends using -Filter and relative comparisons.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Gets sessions associated with the specified tenant.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UntrustedUserName

Gets sessions by the untrusted user name reported directly from the machine.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFullName

Gets sessions by user's full name (usually 'first-name last-name').

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserName

Get sessions by user name (in the form DOMAIN\user).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserSID

Get sessions by user's Windows SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserUPN

Gets sessions by user's User Principal Name (in the form user@domain).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets sessions hosted by machines located in the zone with the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Gets sessions hosted by machines located in the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationUid

Gets sessions running the application with the specified Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SharedDesktopUid

Gets sessions by SharedDesktop Uid.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Broker_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell style filter expression. See about_Broker_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GroupInfo

Each GroupInfo object represents one group, and contains the following properties:

- Count: The count of sessions in this group.
- Name: The value of the property the sessions were grouped by (as a string).

If you do not specify -SortBy, groups are sorted with the largest count first.

Notes

To compare dates or times, use -Filter and relative comparisons. For more information, see about_Broker_Filtering and the examples.

Related Links

- about_Broker_Filtering
- Get-BrokerSession
- Group-Object

Import-BrokerDesktopPolicy

March 11, 2024

Sets the site wide Citrix Group Policy settings for the site.

Syntax

```
Import-BrokerDesktopPolicy
       [-Policy] <Byte[]>
       [-IsBlobOnly]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Import-BrokerDesktopPolicy sets the site wide Citrix Group Policy settings. A successful call to this cmdlet will result in the supplied data being uploaded to every machine in the site prior to its next session launch.

Examples

EXAMPLE 1

This command sets the Citrix Group Policy settings in the site. These policy settings are then applied to every machine prior to the next session launch.


```
Import-BrokerDesktopPolicy $policyData
```

Parameters

-Policy

The configuration data containing the Citrix Group Policy settings to apply to every machine in the site.

Type:	Byte[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-IsBlobOnly

This parameter is reserved for future use.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Byte[]

The configuration data as an opaque binary blob.

Outputs

None

By default, this cmdlet returns no output.

Notes

Import-BrokerDesktopPolicy performs a specialized operation. Direct usage of it in scripts is discouraged, and could result in data corruption. It is recommended that this operation be performed via the Citrix Studio.

Related Links

- Export-BrokerDesktopPolicy
- New-BrokerConfigurationSlot
- New-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

Import-BrokerPolicyTemplates

March 11, 2024

Sets the site wide Citrix Group Policy templates for the site.

Syntax

```
Import-BrokerPolicyTemplates
      [-Templates] <Byte[]>
      [-Version] <Int32>
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Import-BrokerPolicyTemplates sets the site wide Citrix Group Policy templates. A read of the policy templates data using the Export-BrokerPolicyTemplates command should have been executed prior to issuing this command. That command returns a version number that indicates the version of the data stored in the database. If the version number is specified and the number does not match the number stored in the database, it means another import has been executed. In this case, this import will overwrite the changes made by the other import.

Examples

EXAMPLE 1

This command sets the Citrix Group Policy templates in the site.

```
Import-BrokerPolicyTemplates $templatesData 100
```


EXAMPLE 2

This command sets the Citrix Group Policy templates in the site even if the data in the database is not at version 100.

```
1  Import-BrokerPolicyTemplates $templatesData 100 -Force
```

Parameters

-Templates

The configuration data containing the Citrix Group Policy templates.

Type:	Byte[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Version

The current version of the group policy templates data. This number should be obtained from a previous Export-BrokerPolicyTemplates. This number is used to ensure that the data stored in the database has not changed between the last time the data was read and this import. If -Force is not specified, the import will be rejected if the version does not match the version in the database. If -Force is specified, the data stored in the database will be overwritten with the templates provided. The version number always increases by 1 for each write, even when the data is overwritten.

Type:	Int32
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

The current version of the group policy templates data. This number should be obtained from a previous Export-BrokerPolicyTemplates. This number is used to ensure that the data stored in the database has not changed between the last time the data was read and this import. If this parameter is not specified, the templates stored in the database is replaced regardless if there has been another import since the last time the data was exported.

Type:	SwitchParameter
Position:	4
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Byte[]

The configuration data as an opaque binary blob.

System.Int

The current version of the group policy templates data.

Outputs

None

By default, this cmdlet returns no output.

Notes

Import-BrokerPolicyTemplates performs a specialized operation. Direct usage of it in scripts is discouraged, and could result in data corruption. It is recommended that this operation be performed via the Citrix Studio.

Related Links

- Export-BrokerPolicyTemplates

Move-BrokerAdminFolder

March 11, 2024

Moves a folder to another place in the hierarchy, optionally renaming it

Syntax

```
Move-BrokerAdminFolder
    [-InputObject] <AdminFolder[]>
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-BrokerAdminFolder
    [-Name] <String>
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Move-BrokerAdminFolder cmdlet moves a folder for organising objects for administration purposes (for example, Applications) to another position in the hierarchy.

The following special characters are not allowed in the new FolderName: \ / ; : # . * ? = < > | [ ] ( ) “‘

Examples

EXAMPLE 1

Moves the folder called XXX within the folder F1\ to a new home in F2\

```
Move-BrokerAdminFolder F1\XXX\ F2\
```

EXAMPLE 2

Moves the folder called XXX within the folder F1\ to a new home in F2\ renaming it to YYY in the process

```
Move-BrokerAdminFolder F1\XXX\ F2\ -NewName YYY
```


Parameters

-InputObject

The folder(s) to be moved

Type:	AdminFolder[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

A pattern matching the names of folders to be moved

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Destination

The destination folder the folder being moved should end up in

Type:	AdminFolder
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-NewName

The name the new folder should have in the destination folder

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

None or Citrix.Broker.Admin.SDK.AdminFolder

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AdminFolder object.

Related Links

- Get-BrokerAdminFolder
- New-BrokerAdminFolder

Move-BrokerApplication

March 11, 2024

Move a published application from one admin folder to another

Syntax

```
Move-BrokerApplication
    [-InputObject] <Application[]>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-BrokerApplication
    [-Name] <String>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Move-BrokerApplication cmdlet moves a published application from one place to another in the tree of admin folders, optionally renaming it in the process (if you only want to change the name of the application for administrative purposes and not its location in the tree, use the Rename-BrokerApplication cmdlet).

The location and name of a published application in this sense is only of interest to the administrator, changes do not affect the end-user experience.

Examples

EXAMPLE 1

Moves the application in the root folder called “App1”to the folder “F1”.

```
Move-BrokerApplication -Name 'App1' -Destination 'F1\'
```


EXAMPLE 2

Moves the application in folder “F1”called “App1”to the folder “F2”, renaming it to “Application1”in the process.

```
Move-BrokerApplication 'F1\App1' 'F2\' -NewName 'Application1'
```

Parameters

-InputObject

The application(s) to be moved

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The application(s) to be moved

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Destination

The destination location within the admin folder hierarchy

Type:	AdminFolder
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewName

The new name of the application in its new destination

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Application

You can pipe applications to Move-BrokerApplication.

Outputs

None or Citrix.Broker.Admin.SDK.Application

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Application object.

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Add-BrokerApplication
- Get-BrokerApplication
- Remove-BrokerApplication
- Rename-BrokerApplication
- Set-BrokerApplication

Move-BrokerApplicationGroup

March 11, 2024

Move an application group from one admin folder to another

Syntax

```
Move-BrokerApplicationGroup
    [-InputObject] <ApplicationGroup[]>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-BrokerApplicationGroup
    [-Name] <String>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Move-BrokerApplicationGroup cmdlet moves an application group from one place to another in the tree of admin folders, optionally renaming it in the process (if you only want to change the name of the application group for administrative purposes and not its location in the tree, use the Rename-BrokerApplicationGroup cmdlet).

The location and name of an application group in this sense is only of interest to the administrator, changes do not affect the end-user experience.

Examples

Parameters

-InputObject

The application group(s) to be moved

Type:	ApplicationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the application group to be moved

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Destination

The destination location within the admin folder hierarchy.

Type:	AdminFolder

Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewName

The new name of the application group in its new destination

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ApplicationGroup

You can pipe application groups to Move-BrokerApplicationGroup.

Outputs

None or Citrix.Broker.Admin.SDK.ApplicationGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.ApplicationGroup object.

Related Links

- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Rename-BrokerApplicationGroup

Move-BrokerCatalog

March 11, 2024

Move a catalog from one admin folder to another

Syntax

```
Move-BrokerCatalog
    [-InputObject] <Catalog[]>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-BrokerCatalog
    [-Name] <String>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Move-BrokerCatalog cmdlet moves a catalog from one place to another in the tree of admin folders, optionally renaming it in the process (if you only want to change the name of the catalog for administrative purposes and not its location in the tree, use the Rename-BrokerCatalog cmdlet).

The location and name of a catalog in this sense is only of interest to the administrator, changes do not affect the end-user experience.

Examples

Parameters

-InputObject

The catalog(s) to be moved

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the catalog to be moved

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Destination

The destination location within the admin folder hierarchy.

Type:	AdminFolder
Position:	3
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-NewName

The new name of the catalog in its new destination

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

You can pipe catalogs to Move-BrokerCatalog.

Outputs

None or Citrix.Broker.Admin.SDK.Catalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Catalog object.

Related Links

- about_Broker_RemotePC
- Get-BrokerCatalog

- New-BrokerCatalog
- Remove-BrokerCatalog
- Rename-BrokerCatalog

Move-BrokerDesktopGroup

March 11, 2024

Move a desktop group from one admin folder to another

Syntax

```
Move-BrokerDesktopGroup
    [-InputObject] <DesktopGroup[]>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-BrokerDesktopGroup
    [-Name] <String>
    [-PassThru]
    [-Destination] <AdminFolder>
    [-NewName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Move-BrokerDesktopGroup cmdlet moves a desktop group from one place to another in the tree of admin folders, optionally renaming it in the process (if you only want to change the name of the desktop group for administrative purposes and not its location in the tree, use the Rename-BrokerDesktopGroup cmdlet).

The location and name of a desktop group in this sense is only of interest to the administrator, changes do not affect the end-user experience.

Examples

Parameters

-InputObject

The desktop group(s) to be moved

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the desktop group to be moved

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Destination

The destination location within the admin folder hierarchy.

Type:	AdminFolder
Position:	3
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-NewName

The new name of the desktop group in its new destination

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

You can pipe desktop groups to Move-BrokerDesktopGroup.

Outputs

None or Citrix.Broker.Admin.SDK.DesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.DesktopGroup object.

Related Links

- Get-BrokerDesktopGroup
- New-BrokerDesktopGroup

- Remove-BrokerDesktopGroup
- Rename-BrokerDesktopGroup

Move-BrokerGpoPolicy

March 11, 2024

Move policies to another policy set. The source policies must be in the same policy set. They must all not exist in the target policy set. An error will be reported if there is a name conflict and the move will fail. When the move fails, no policy is moved.

Syntax

```
Move-BrokerGpoPolicy
    -TargetPolicySet <Guid>
    -PolicyGuids <Guid[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

If the target policy set is not specified, the policies are deleted and not moved to any other policy set.

Examples

EXAMPLE 1

Copy policies to the target policy set.

```
Copy-BrokerGpoPolicy -TargetPolicySet "abcd1234-..." -PolicyGuids @("
    abcdef12-...", "12345678-...")
```

Parameters

-TargetPolicySet

GUID of the target policy set

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicyGuids

GUIDs of the policies to be moved

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

New-BrokerAccessPolicyRule

March 11, 2024

Creates a new rule in the site's access policy.

Syntax

```
New-BrokerAccessPolicyRule
   [-AllowedConnections <AllowedConnection>]
   [-AllowedProtocols <String[]>]
   [-AllowedUsers <AllowedUser>]
   [-AllowRestart <Boolean>]
   [-AppProtectionKeyLoggingRequired <Boolean>]
```



```
    [-AppProtectionScreenCaptureRequired <Boolean>]
    [-Description <String>]
    -DesktopGroupUid <Int32>
    [-Enabled <Boolean>]
    [-ExcludedClientIPFilterEnabled <Boolean>]
    [-ExcludedClientIPs <IPAddressRange[]>]
    [-ExcludedClientNameFilterEnabled <Boolean>]
    [-ExcludedClientNames <String[]>]
    [-ExcludedSmartAccessFilterEnabled <Boolean>]
    [-ExcludedSmartAccessTags <String[]>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-HdxSslEnabled <Boolean>]
    [-IncludedClientIPFilterEnabled <Boolean>]
    [-IncludedClientIPs <IPAddressRange[]>]
    [-IncludedClientNameFilterEnabled <Boolean>]
    [-IncludedClientNames <String[]>]
    [-IncludedSmartAccessFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterType <String>]
    [-IncludedSmartAccessTags <String[]>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-Name] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-BrokerAccessPolicyRule
    [-AllowedConnections <AllowedConnection>]
    [-AllowedProtocols <String[]>]
    [-AllowedUsers <AllowedUser>]
    [-AllowRestart <Boolean>]
    [-AppProtectionKeyLoggingRequired <Boolean>]
    [-AppProtectionScreenCaptureRequired <Boolean>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-ExcludedClientIPFilterEnabled <Boolean>]
    [-ExcludedClientIPs <IPAddressRange[]>]
    [-ExcludedClientNameFilterEnabled <Boolean>]
    [-ExcludedClientNames <String[]>]
    [-ExcludedSmartAccessFilterEnabled <Boolean>]
    [-ExcludedSmartAccessTags <String[]>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-HdxSslEnabled <Boolean>]
    [-IncludedClientIPFilterEnabled <Boolean>]
    [-IncludedClientIPs <IPAddressRange[]>]
    [-IncludedClientNameFilterEnabled <Boolean>]
    [-IncludedClientNames <String[]>]
    [-IncludedDesktopGroupFilterEnabled <Boolean>]
    -IncludedDesktopGroups <DesktopGroup[]>
    [-IncludedSmartAccessFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterType <String>]
```



```
    [-IncludedSmartAccessTags <String[]>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-Name] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerAccessPolicyRule cmdlet adds a new rule to the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Multiple rules in the access policy can apply to the same desktop group.

For a user to gain access to a desktop group via a rule their connection must match all its enabled include filters, and none of its enabled exclude filters. In addition, for a user to be able to launch a desktop or application resource session from the desktop group, they must have an entitlement to use the resource granted by the entitlement or assignment policies, or by direct machine assignment.

Examples

EXAMPLE 1

Creates an access policy rule allowing access to the Tech Support desktop group for all users of the SUPPORT\uk-staff group. Connections to desktop or application resources in the group can only be made using the HDX protocol.

For users to gain access to resources in the group also requires that, depending on the desktop kind of the group, appropriate assignment or entitlement policy rules, or explicit machine assignments exist.

```
$dg = Get-BrokerDesktopGroup 'Tech Support'
New-BrokerAccessPolicyRule 'UK Tech Support' -IncludedUserFilterEnabled
    $true -IncludedUsers support\uk-staff -DesktopGroupUid $dg.Uid -
   AllowedProtocols 'HDX'
```


Parameters

-Name

Specifies the administrative name of the new rule. Each rule within the site's access policy must have a unique name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

Specifies the desktop group to which the new rule applies.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedDesktopGroups

This parameter is supported for backward compatibility only. If used only a single desktop group UID can be specified.

The IncludedDesktopGroups and IncludedDesktopGroupFilterEnabled parameters have been superseded by the DesktopGroupUid parameter.

Type:	DesktopGroup[]
Position:	Named

Default value:	(empty list)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AllowedConnections

Specifies whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, ViaAG and AnyViaAG.

For a detailed description of this property see "help about_Broker_AccessPolicy".

Type:	AllowedConnection
Position:	Named
Default value:	Filtered
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AllowedProtocols

Specifies the protocols (for example HDX, RDP) available to the user for sessions delivered from the new rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.

If the protocol list is empty, access to the desktop group is implicitly denied.

Type:	String[]
Position:	Named
Default value:	HDX
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-AllowedUsers

Specifies the behavior of the included users filter of the new rule. This can restrict access to a list of named users or groups, allow access to any authenticated user, any user (whether authenticated or not), or only non-authenticated users. For a detailed description of this property see "help about_Broker_AccessPolicy".

Valid values are Filtered, AnyAuthenticated, Any, AnonymousOnly and FilteredOrAnonymous.

Type:	AllowedUser
Position:	Named
Default value:	Filtered
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AllowRestart

Specifies if the user can restart sessions delivered from the new rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppProtectionKeyLoggingRequired

Specifies whether key logging app protection is required.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppProtectionScreenCaptureRequired

Specifies whether screen capture app protection is required.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies an optional description of the new rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new rule is initially enabled. A disabled rule is ignored when evaluating the site's access policy.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedClientIPFilterEnabled

Specifies whether the excluded client IP address filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedClientIPs

Specifies IP addresses of user devices explicitly denied access to the new rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the excluded client IP address filter.

Type:	IPAddressRange[]
Position:	Named
Default value:	(empty list)
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedClientNameFilterEnabled

Specifies whether the excluded client names filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedClientNames

Specifies names of user devices explicitly denied access to the new rule's desktop group. This property forms part of the excluded client names filter.

Type:	String[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedSmartAccessFilterEnabled

Specifies whether the excluded SmartAccess tags filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedSmartAccessTags

Specifies SmartAccess tags which explicitly deny access to the new rule's desktop group if any occur in those provided with the user's connection. This property forms part of the excluded SmartAccess tags filter.

Type:	String[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Specifies whether the excluded users filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUsers

Specifies any users and groups who are explicitly denied access to the new rule's desktop group. This property forms part of the excluded users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-HdxSslEnabled

Indicates whether TLS encryption is enabled for sessions delivered from the rule's desktop group.

Type:	Boolean
Position:	Named
Default value:	$false
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedClientIPFilterEnabled

Specifies whether the included client IP address filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-IncludedClientIPs

Specifies IP addresses of user devices allowed access to the new rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

Type:	IPAddressRange[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedClientNameFilterEnabled

Specifies whether the included client name filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedClientNames

Specifies names of user devices allowed access to the new rule's desktop group. This property forms part of the included client names filter.

Type:	String[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedSmartAccessFilterEnabled

Specifies whether the included SmartAccess tags filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedSmartAccessFilterType

Specifies whether all tags present in IncludedSmartAccessTags must match tags provided by the user's connection to grant access (MatchAll), or whether any tag matching is sufficient (MatchAny).

Type:	String
Accepted values:	MatchAll, MatchAny
Position:	Named
Default value:	MatchAny
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedSmartAccessTags

Specifies SmartAccess tags which grant access to the new rule's desktop group if they occur in those provided with the user's connection. If multiple tags are specified, access also depends on the IncludedSmartAccessFilterType setting. This property forms part of the included SmartAccess tags filter.

Type:	String[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Specifies whether the included users filter is initially enabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUsers

Specifies users and groups who are granted access to the new rule's desktop group. This property forms part of the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedDesktopGroupFilterEnabled

This parameter is supported for backward compatibility only. If used the supplied value must be $true.

The IncludedDesktopGroups and IncludedDesktopGroupFilterEnabled parameters have been superseded by the DesktopGroupUid parameter.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AccessPolicyRule

New-BrokerAccessPolicyRule returns the newly created access policy rule.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- Get-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule

New-BrokerAdminFolder

March 11, 2024

Creates a new admin folder.

Syntax

```
New-BrokerAdminFolder
   [-FolderName] <String>
   [-ParentFolder <AdminFolder>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerAdminFolder cmdlet creates a new folder for organising objects for administration purposes (for example, Applications).

New-BrokerAdminFolder creates the folder object and optionally places it within an existing admin folder if required.

The following special characters are not allowed in the FolderName: \ / ; : # . * ? = < > | [ ] ( ) “‘

Examples

EXAMPLE 1

Creates an admin folder called F1 under the root folder (i.e. F1\)

```
New-BrokerAdminFolder F1
```

EXAMPLE 2

Creates an admin folder called F2 under the folder F1\ (i.e. F1\F2\)

```
New-BrokerAdminFolder F2 -AdminFolder F1\
```

Parameters

-FolderName

The simple name of the new folder within its parent (if any)

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ParentFolder

The name or UID of the parent folder (if any)

Type:	AdminFolder
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

Citrix.Broker.Admin.SDK.AdminFolder

The new admin folder.

Related Links

- Get-BrokerAdminFolder
- Remove-BrokerAdminFolder

New-BrokerAppAssignmentPolicyRule

March 11, 2024

Creates a new application rule in the site's assignment policy.

Syntax

```
New-BrokerAppAssignmentPolicyRule
   [-Description <String>]
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-Name] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerAppAssignmentPolicyRule cmdlet adds a new application rule to the site's assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

The following constraints apply when creating an application assignment rule for a desktop group:

- The group's desktop kind must be Private
- The group's delivery type must be AppsOnly
- Only a single application rule can apply to a given group
- Application assignment rules cannot be applied to RemotePC groups.

When a user selects an application published from a private group, a currently unassigned machine is selected from the group and permanently assigned to the user. An application session is then launched to the machine. Subsequent launches are routed directly to the now assigned machine.

Once a machine has been assigned in this way, the original assignment rule plays no further part in access to the machine.

Examples

EXAMPLE 1

Creates an application rule in the assignment policy that grants all members of the SALES\uk-staff group an entitlement to a single machine from the Sales Support desktop group. The machine can be used for running applications published from the group.


```
$dg = Get-BrokerDesktopGroup 'Sales Support'
New-BrokerAppAssignmentPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid
    -IncludedUsers sales\uk-staff
```

Parameters

-Name

Specifies the administrative name of the new application rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

Specifies the unique ID of the desktop group to which the new application rule applies.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies an optional description of the new application rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new application rule is initially enabled. A disabled rule is ignored when evaluating the site's assignment policy.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUsers

Specifies the excluded users filter of the new application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the new application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUsers

Specifies the included users filter of the new application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

New-BrokerAppAssignmentPolicyRule returns the newly created application rule in the assignment policy.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- Get-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule
- Remove-BrokerAppAssignmentPolicyRule

New-BrokerAppEntitlementPolicyRule

March 11, 2024

Creates a new application rule in the site's entitlement policy.

Syntax

```
New-BrokerAppEntitlementPolicyRule
   [-Description <String>]
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-LeasingBehavior <LeasingBehavior>]
   [-Name] <String>
```



```
    [-SessionReconnection <SessionReconnection>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerAppEntitlementPolicyRule cmdlet adds a new application rule to the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

The following constraints apply when creating an application entitlement rule for a desktop group:

- The group's desktop kind must be Shared
- The group's delivery type must be AppsOnly or DesktopsAndApps
- Only a single application rule can apply to a given group

When a user selects an application published from a shared group, a machine is selected from the group on which to run the application. No permanent association exists between the user and the selected machine; once the session ends the association also ends.

Even though only a single application entitlement and therefore session can be defined for a group, the user can still run multiple applications from the group because the applications run within the same session.

Examples

EXAMPLE 1

Creates an application rule in the entitlement policy that entitles all members of the SUPPORT\uk-staff group to a machine for running applications published from the Customer Support desktop group.

```
$dg = Get-BrokerDesktopGroup 'Customer Support'
New-BrokerAppEntitlementPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid
    -IncludedUsers support\uk-staff
```


Parameters

-Name

Specifies the administrative name of the new application rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

Specifies the unique ID of the desktop group to which the new application rule applies.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies an optional description of the new application rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	Null
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new application rule is initially enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUsers

Specifies the excluded users filter of the application rule, that is, the users and groups who are explicitly denied entitlements to published applications from the desktop group.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the new rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUsers

Specifies the included users filter of the application rule, that is, the users and groups who are granted an entitlement to an application session by the new rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named

Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LeasingBehavior

Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:

Allowed and Disallowed.

The Allowed value indicates that connection leasing should behave normally. The Disallowed value prevents users

from launching or reconnecting to sessions using this entitlement while connection leasing is active (typically during a database outage).

Type:	LeasingBehavior
Position:	Named
Default value:	Allowed
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SessionReconnection

Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	Always
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

New-BrokerAppEntitlementPolicyRule returns the newly created application rule in the entitlement policy.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- Get-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule

New-BrokerApplication

March 11, 2024

Creates a new published application.

Syntax

```
New-BrokerApplication
   [-AdminFolder <AdminFolder>]
   [-ApplicationType <ApplicationType>]
   [-BrowserName <String>]
   [-ClientFolder <String>]
   [-CommandLineArguments <String>]
   -CommandLineExecutable <String>
   [-CpuPriorityLevel <CpuPriorityLevel>]
   [-Description <String>]
   -DesktopGroup <DesktopGroup>
   [-DoNotEnumerate <Boolean>]
   [-Enabled <Boolean>]
   [-HomeZoneOnly <Boolean>]
   [-HomeZoneUid <Guid>]
   [-IconFromClient <Boolean>]
   [-IconUid <Int32>]
   [-IgnoreUserHomeZone <Boolean>]
   [-LocalLaunchDisabled <Boolean>]
   [-MaxPerMachineInstances <Int32>]
```



```
    [-MaxPerUserInstances <Int32>]
    [-MaxTotalInstances <Int32>]
    [-Name] <String>
    [-PackagedApplicationId <String>]
    [-PackagedApplicationType <String>]
    [-Priority <Int32>]
    [-PublishedName <String>]
    [-SecureCmdLineArgumentsEnabled <Boolean>]
    [-ShortcutAddedToDesktop <Boolean>]
    [-ShortcutAddedToStartMenu <Boolean>]
    [-StartMenuFolder <String>]
    [-UserFilterEnabled <Boolean>]
    [-UUID <Guid>]
    [-Visible <Boolean>]
    [-WaitForPrinterCreation <Boolean>]
    [-WorkingDirectory <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-BrokerApplication
    [-AdminFolder <AdminFolder>]
    -ApplicationGroup <ApplicationGroup>
    [-ApplicationType <ApplicationType>]
    [-BrowserName <String>]
    [-ClientFolder <String>]
    [-CommandLineArguments <String>]
    -CommandLineExecutable <String>
    [-CpuPriorityLevel <CpuPriorityLevel>]
    [-Description <String>]
    [-DoNotEnumerate <Boolean>]
    [-Enabled <Boolean>]
    [-HomeZoneOnly <Boolean>]
    [-HomeZoneUid <Guid>]
    [-IconFromClient <Boolean>]
    [-IconUid <Int32>]
    [-IgnoreUserHomeZone <Boolean>]
    [-LocalLaunchDisabled <Boolean>]
    [-MaxPerMachineInstances <Int32>]
    [-MaxPerUserInstances <Int32>]
    [-MaxTotalInstances <Int32>]
    [-Name] <String>
    [-PackagedApplicationId <String>]
    [-PackagedApplicationType <String>]
    [-PublishedName <String>]
    [-SecureCmdLineArgumentsEnabled <Boolean>]
    [-ShortcutAddedToDesktop <Boolean>]
    [-ShortcutAddedToStartMenu <Boolean>]
    [-StartMenuFolder <String>]
    [-UserFilterEnabled <Boolean>]
    [-UUID <Guid>]
    [-Visible <Boolean>]
    [-WaitForPrinterCreation <Boolean>]
```



```
    [-WorkingDirectory <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerApplication cmdlet creates a new published application in the site.

New-BrokerApplication creates the application object, and associates it with a desktop group or application group. Application objects have three names that identify them (in addition to their Uid): the Name, BrowserName and the PublishedName. The BrowserName is unique across the entire site, and is primarily used internally. The Name is also unique and is what is seen by the administrator; it contains any prefix for an enclosing admin folder (if any). The PublishedName is not unique and is what is seen by the users.

You can create HostedOnDesktop, InstalledOnClient or PublishedContent applications but the ApplicationType cannot be changed later.

The following special characters are not allowed in the Name, BrowserName or the PublishedName properties: \ / ; : # . * ? = < > | [ ] ( ) “

In addition the ‘ character is not allowed in the Name property.

See about_Broker_Applications for more information.

Examples

EXAMPLE 1

Creates and returns an object for a published application called “Notepad”that launches “notepad.exe”.

```
New-BrokerApplication -ApplicationType HostedOnDesktop -Name "Notepad"
    -CommandLineExecutable "notepad.exe" -DesktopGroup PrivateDG1
```

EXAMPLE 2

Creates and returns an object for a published application called “Citrix.com”that launches the URL https://www.citrix.com/.

```
New-BrokerApplication -ApplicationType PublishedContent -Name "Citrix.
    com" -CommandLineExecutable "https://www.citrix.com/" -DesktopGroup
    SharedDG1
```


EXAMPLE 3

This is a much more complete example. It creates an application object to publish Notepad and associates it first with the "SharedDG1"desktop group.

Next it adds an additional desktop group (one that can host applications), and publishes the application to that desktop group. It then gets the ImportedFTA object for the .txt file-type extension (this assumes file-type associations have already been imported), and then configures it so that ".txt"is associated with the published application.

Note: The appropriate access policy and app assignment/entitlement rules must also be configured to allow access to the application.

```
1 $dg = Get-BrokerDesktopGroup "SharedDG1"
2 $app = New-BrokerApplication -ApplicationType HostedOnDesktop -Name "
     Notepad" -CommandLineExecutable "notepad.exe" -DesktopGroup $dg
3 $group = Get-BrokerDesktopGroup -Name "Shared desktop group"
4 Add-BrokerApplication $app -DesktopGroup $group
5 $fta = Get-BrokerImportedFTA -ExtensionName ".txt"
6 New-BrokerConfiguredFTA -ImportedFTA $fta -ApplicationUid $app.Uid
```

Parameters

-Name

Specifies the name of the application (must be unique within folder).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CommandLineExecutable

Specifies the name of the executable file to launch. The full path need not be provided if it's already in the path. Environment variables can also be used.

Type:	String

Position:	Named
Default value:	(required)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ApplicationGroup

Specifies which application group this application should be associated with. Associations between applications and desktop groups or application groups can be added or removed using the Add-BrokerApplication and Remove-BrokerApplication cmdlets.

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DesktopGroup

Specifies which desktop group this application should be associated with. Associations between applications and desktop groups or application groups can be added or removed using the Add-BrokerApplication and Remove-BrokerApplication cmdlets.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AdminFolder

The folder in which the new application should reside (if any).

Type:	AdminFolder
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ApplicationType

Specifies the type of the application: HostedOnDesktop, InstalledOnClient or PublishedContent.

Type:	ApplicationType
Position:	Named
Default value:	(required)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-BrowserName

Specifies the internal name for this application. It must be unique in the site.

Type:	String
Position:	Named
Default value:	(same as Name)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ClientFolder

Specifies the folder that the application belongs to as the user sees it. This is the application folder that is seen in the Citrix Online Plug-in, in Web Services, and also in the end-user's Start menu. Sub-directories can be specified with '' character. The following special characters are not allowed: / * ? < > | ":. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CommandLineArguments

Specifies the command-line arguments to use when launching the executable. Environment variables can be used. This setting is ignored for applications of type PublishedContent.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CpuPriorityLevel

Specifies the CPU priority for the launched process. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	CpuPriorityLevel
Position:	Named

Default value:	Normal
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies the description of the application. This is only seen by Citrix administrators and is not visible to users.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DoNotEnumerate

Specifies if the application is returned to the user by enumeration.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Specifies whether or not this application can be launched.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-HomeZoneOnly

Specifies whether if the preferred zone for launching the application is its home zone but no machine is available from that zone then the launch fails.

This can only be set if the application has a home zone preference specified.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-HomeZoneUid

Specifies any home zone preference used when launching this application.

Type:	Guid
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IconFromClient

Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IconUid

Specifies which icon to use for this application. This icon is visible both to the administrator (in the consoles) and to the user. If no icon is specified, then a generic built-in application icon is used.

Type:	Int32
Position:	Named
Default value:	2
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IgnoreUserHomeZone

Specifies that when launching the application and the user has a home zone specified then the user's home zone preference should be ignored.

This can only be set if the application does not itself have a home zone preference specified.

Type:	Boolean
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LocalLaunchDisabled

When launching a published application from within a published desktop, do not launch the application in that desktop session.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxPerMachineInstances

Specifies the maximum allowed concurrently running instances of the application that an individual machine can have. A value of zero allows unlimited usage subject to any site-wide limit.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxPerUserInstances

Specifies the maximum allowed concurrently running instances of the application that an individual user can have. A value of zero allows unlimited usage subject to any site-wide limit.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxTotalInstances

Specifies the maximum allowed total of concurrently running instances of the application in the site. A value of zero allows unlimited usage.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PackagedApplicationId

The Id of the Packaged Application in the AppLibrary

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PackagedApplicationType

The packaging technology used to create this application

Type:	String
Accepted values:	AppAttach, AppVDualAdmin, AppVSingleAdmin, FlexApp, Msix, NotApplicable
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PublishedName

The name seen by end users who have access to this application.

Type:	String
Position:	Named
Default value:	The same value as that supplied for the name of the application.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-SecureCmdLineArgumentsEnabled

Specifies whether the command-line arguments are secured or not. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ShortcutAddedToDesktop

Specifies whether or not a shortcut to the application should be placed on the user device. This is valid only for the Citrix Online Plug-in.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ShortcutAddedToStartMenu

Specifies whether a shortcut to the application should be placed in the user's start menu on their user device.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-StartMenuFolder

Specifies the name of the start menu folder that holds the application shortcut (if any). This is valid only for the Citrix Online Plug-in. Subdirectories can be specified with ' character. The following special characters are not allowed: / * ? < > | ":.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-UserFilterEnabled

Specifies whether the application's user filter is enabled or disabled. Where the user filter is enabled, the application is visible only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

An optional GUID for this application.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Visible

Specifies whether or not this application is visible to users. Note that it's possible for an application to be disabled and still visible.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WaitForPrinterCreation

Specifies whether or not the session waits for the printers to be created before allowing the user to interact with the session. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WorkingDirectory

Specifies which working directory the executable is launched from. Environment variables can be used. This setting is ignored for applications of type PublishedContent.

Type:	String
Position:	Named
Default value:	Null
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Priority

Specifies the priority of the mapping between the application and desktop group. A value of zero has the highest priority, with increasing values indicating lower priorities.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

Citrix.Broker.Admin.SDK.Application

New-BrokerApplication returns an Application object.

Notes

Usually only the Name is specified with the New-BrokerApplication cmdlet, and the system chooses a BrowserName and PublishedName for you. By default the BrowserName is the same as the Name, if it is unique in the site. If not, then "-x"is appended to the name, where "x"is a number. For instance, if there is already an application with a BrowserName of "Notepad"and a new application is created with a Name of "Notepad", then the new application gets a BrowserName of "Notepad-1". If another "Notepad"is published, it has a BrowserName of "Notepad-2".

That said, the BrowserName can optionally be specified as well.

Related Links

- about_Broker_Applications
- Add-BrokerApplication
- Remove-BrokerApplication
- Get-BrokerApplication
- Remove-BrokerApplication
- Rename-BrokerApplication
- Move-BrokerApplication
- Set-BrokerApplication

New-BrokerApplicationGroup

March 11, 2024

Create a new application group to which applications can be added.

Syntax

```
New-BrokerApplicationGroup
   [-AdminFolder <AdminFolder>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-Name] <String>
   [-RestrictToTag <String>]
   [-Scope <String[]>]
   [-SessionSharingEnabled <Boolean>]
   [-SingleAppPerSession <Boolean>]
   [-TenantId <Guid>]
   [-UserFilterEnabled <Boolean>]
   [-UUID <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerApplicationGroup cmdlet creates a new application group. Applications that are added to the application group can then be managed centrally by setting properties on the application group rather than on each application individually.

Application groups may also be used to isolate applications that should not share sessions with other applications. To do this, create an application group with SessionSharingEnabled equal to $false, and then add to it those applications that you wish to isolate. The isolated applications continue to share sessions with each other, but not with any other published applications.

To create a new application and add it to an application group, use New-BrokerApplication -ApplicationGroup. To add an existing application to an application group, use Add-BrokerApplication -ApplicationGroup.

After adding applications to an application group, you must then publish the application group to a desktop group before its applications can be launched. Use the Add-BrokerApplicationGroup cmdlet to do this.

To manipulate the user filter associated with an application group, use Add-BrokerUser -ApplicationGroup and Remove-BrokerUser -ApplicationGroup.

To manipulate the set of tags associated with an application group, use Add-BrokerTag -ApplicationGroup and Remove-BrokerTag -ApplicationGroup.

See about_Broker_Applications for more information.

Examples

EXAMPLE 1

Creates a new application group called 'Helpdesk Apps'.

```
New-BrokerApplicationGroup "Helpdesk Apps"
```

EXAMPLE 2

Creates a new application group called 'Accounts Apps', and then restrict access so that only members of the MYDOMAIN\Accounts user group can launch applications in 'Accounts Apps'.

```
New-BrokerApplicationGroup "Accounts Apps" -UserFilterEnabled $true
Add-BrokerUser "MYDOMAIN\Accounts" -ApplicationGroup "Accounts Apps"
```

Parameters

-Name

A name for the application group. Not visible to end users.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AdminFolder

The folder in which the new application group should reside (if any).

Type:	AdminFolder
Position:	Named
Default value:	None (root folder)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description for the application group. Not visible to end users.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Whether the application group's applications can be launched by end users.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RestrictToTag

Optional tag that may be used further to restrict which machines may be used for launching the application group's applications. A machine may be used by an application group if either the application group has no tag restriction or the application group does have a tag restriction and the machine is tagged with the same tag.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Scope

Specifies the name of the delegated administration scope to which the application group should belong.

Type:	String[]
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SessionSharingEnabled

Whether the application group's applications can share sessions with applications that are not a member of this application group. Please note this setting and SingleAppPerSession cannot be true at the same time.

Type:	Boolean
Position:	Named

Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SingleAppPerSession

Specifies whether each application launched from this application group starts in its own new session or can share an existing suitable session if present. Please note this setting and SessionSharingEnabled cannot be true at the same time.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with application group. Must always be specified in multi-tenant sites, must not be specified otherwise.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UserFilterEnabled

Whether the application group's user filter is enabled or disabled. Where the user filter is enabled, the application is visible only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

The UUID of the application group. If a UUID is not provided, then one will be generated automatically.

Type:	Guid
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ApplicationGroup

The newly created application group.

Related Links

- about_Broker_Applications
- Add-BrokerApplicationGroup
- Get-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Rename-BrokerApplicationGroup
- Move-BrokerApplicationGroup

- Set-BrokerApplicationGroup
- Add-BrokerApplication
- Remove-BrokerApplication
- Add-BrokerUser
- Remove-BrokerUser
- Add-BrokerTag
- Remove-BrokerTag

New-BrokerAssignmentPolicyRule

March 11, 2024

Creates a new desktop rule in the site's assignment policy.

Syntax

```
New-BrokerAssignmentPolicyRule
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IconUid <Int32>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-MaxDesktops <Int32>]
   [-Name] <String>
   [-PublishedName <String>]
   [-SecureIcaRequired <Boolean>]
   [-UUID <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerAssignmentPolicyRule cmdlet adds a new desktop rule to the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

The following constraints apply when creating a desktop assignment rule for a desktop group:

- The group's desktop kind must be Private
- The group's delivery type must be DesktopsOnly
- Only one desktop assignment rule can be created for RemotePC groups.

When a user selects a machine assignment entitlement from a private group, a currently unassigned machine is selected from the group and permanently assigned to the user to create an assigned desktop. A desktop session is then launched to the machine. Subsequent launches are routed directly to the now assigned machine.

Once a machine has been assigned in this way, the original assignment rule plays no further part in access to the new desktop.

Multiple desktop rules in the assignment policy can apply to the same desktop group. Where a user is granted entitlements by more than one rule for the same group, they can have as many machine assignments from the group as the total of their entitlements.

Examples

EXAMPLE 1

Creates a desktop rule in the assignment policy that grants all members of the SALES\uk-staff group an entitlement to a single machine from the Sales Support desktop group. The entitlement name seen by users is Sales Desktop.

```
1 $dg = Get-BrokerDesktopGroup 'Sales Support'
2 New-BrokerAssignmentPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid -
     IncludedUsers sales\uk-staff -PublishedName 'Sales Desktop'
```

Parameters

-Name

Specifies the administrative name of the new desktop rule. Each rule in the site's assignment policy must have a unique name (irrespective of whether they are desktop or application rules).

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

Specifies the unique ID of the desktop group to which the new desktop rule applies.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ColorDepth

Specifies the color depth of any desktop sessions to machines assigned by the new rule.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	ColorDepth
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies an optional description of the new desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new desktop rule is initially enabled. A disabled rule is ignored when evaluating the site's assignment policy.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUsers

Specifies the excluded users filter of the new desktop rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IconUid

Specifies the unique ID of the icon used to display the machine assignment entitlement to the user, and of the assigned desktop itself following the assignment.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Int32
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the new desktop rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUsers

Specifies the included users filter of the new desktop rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxDesktops

The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.

Type:	Int32
Position:	Named
Default value:	1

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PublishedName

The name of the new machine assignment entitlement as seen by the user, and of the assigned desktop following its usage.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-SecureIcaRequired

Specifies whether the new desktop rule requires the SecureICA protocol to be used for desktop sessions to machines assigned using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Boolean
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

An optional GUID for this rule.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

New-BrokerAssignmentPolicyRule returns the newly created desktop rule in the assignment policy rule.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_ErrorHandling
- Get-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule
- Remove-BrokerAssignmentPolicyRule

New-BrokerAutoTagRule

March 11, 2024

Creates a new AutoTagRule.

Syntax

```
New-BrokerAutoTagRule
   [-Description <String>]
   [-Name] <String>
   -ObjectType <String>
   -RuleText <String>
   -TagUid <Int32>
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Creates an AutoTagRule which defines a rule to select objects to assign (or remove) a Tag to (or from).

Examples

EXAMPLE 1

Creates a new rule for the “MyExample”tag.

```
$tag = New-BrokerTag -Name MyExample
New-BrokerAutoTagRule -Name RandomAllocatedCatalogs -TagUid $tag.Uid -
    ObjectType Catalog -RuleText "-AllocationType Random"
```

Parameters

-Name

The rule name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectType

The object type on which the rule is applied.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RuleText

Rule in the text format.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-TagUid

The Tag Uid this rule applies to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

The Description of rule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.AutoTagRule

Outputs the AutoTagRule object.

Related Links

- about_Broker_AutoTagRule
- Get-BrokerAutoTagRule
- Set-BrokerAutoTagRule
- Remove-BrokerAutoTagRule
- Rename-BrokerAutoTagRule

New-BrokerCatalog

March 11, 2024

Adds a new catalog to the site.

Syntax

```
New-BrokerCatalog
   [-AdminFolder <AdminFolder>]
   [-AllocationType] <AllocationType>
   [-Description <String>]
   [-IsRemotePC <Boolean>]
   [-MachinesArePhysical <Boolean>]
   [-MdmEnrollment <String>]
   [-MinimumFunctionalLevel <FunctionalLevel>]
   [-Name] <String>
   [-PersistUserChanges] <PersistUserChanges>
   [-ProvisioningSchemeId <Guid>]
   [-ProvisioningType] <ProvisioningType>
   [-PvsAddress <String>]
```



```
    [-PvsDomain <String>]
    [-RemotePCHypervisorConnectionUid <Int32>]
    [-Scope <String[]>]
    [-SessionSupport] <SessionSupport>
    [-TenantId <Guid>]
    [-TimeZone <String>]
    [-UUID <Guid>]
    [-ZoneUid <Guid>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-BrokerCatalog
    [-PvsForVM <String[]>]
    [-AdminFolder <AdminFolder>]
    [-AllocationType] <AllocationType>
    [-CatalogKind] <CatalogKind>
    [-Description <String>]
    [-IsRemotePC <Boolean>]
    [-MachinesArePhysical <Boolean>]
    [-MdmEnrollment <String>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-Name] <String>
    [-PvsAddress <String>]
    [-PvsDomain <String>]
    [-RemotePCHypervisorConnectionUid <Int32>]
    [-Scope <String[]>]
    [-TenantId <Guid>]
    [-TimeZone <String>]
    [-UUID <Guid>]
    [-ZoneUid <Guid>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

New-BrokerCatalog adds a catalog through which machines can be provided to the site.

In order for a machine to register in a site, the machine must belong to a catalog with which it is compatible. The compatibility of a machine with a catalog is determined by two of the parameters of New-BrokerCatalog:

- MinimalFunctionalLevel: The minimal functional level supported in the

catalog. The functional level of the machine is determined by the capabilities of the Citrix VDA software on it.

- SessionSupport: The session support (single/multi) of the catalog. The

session support of the machine is determined by the variant of the Citrix VDA software installed (workstation/terminal services, respectively).

Examples

EXAMPLE 1

This command creates a catalog that can contain unmanaged physical or virtual machines that are permanently assigned to the user.

```
New-BrokerCatalog -AllocationType Static -CatalogKind Unmanaged -
    Description "Catalog1 Description" -Name "Catalog1 Name"
```

EXAMPLE 2

This command creates a catalog that can contain power-managed machines that are randomly assigned to the user.

```
New-BrokerCatalog -AllocationType Random -CatalogKind PowerManaged -
    Description "catalog 2 Description" -Name "Catalog2 Name"
```

EXAMPLE 3

This command creates a catalog that can contain managed machines that are provisioned using Provisioning Services.

```
New-BrokerCatalog -AllocationType Random -CatalogKind PVS -Description
    "PVS Catalog Desc" -Name "PVS Catalog Name" -PvsAddress "
    pvsServer@pvsDomain.com" -PvsDomain "pvsDomain.com" -PvsForVM $(
    $farmGuid:$schemeGuid)
```

Parameters

-Name

Specifies a name for the catalog. Each catalog within a site must have a unique name.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AllocationType

Specifies how machines in the catalog are assigned to users. Values can be:

- Static - Machines in a catalog of this type are permanently assigned to a user.
- Permanent - equivalent to 'Static'.
- Random - Machines in a catalog of this type are picked at random and temporarily assigned to a user.

Type:	AllocationType
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-CatalogKind

Deprecated: The type of machines the catalog will contain. Values can be: ThinCloned, SingleImage, PowerManaged, Unmanaged, or Pvs.

- Thin-Cloned, Single-Image and Personal vDisk Catalogs

  Thin-cloned and single-image catalog kinds are for machines created and managed with Provisioning Services for VMs. All machines in this type of catalog are managed, and so must be associated with a hypervisor connection.
  A thin-cloned catalog is used for original golden VM images that are cloned when they are assigned to a VM, and users' changes to the VM image are retained after the VM is restarted.
  A single-image catalog is used when multiple machines provisioned with Provisioning Services for VMs

all share a single golden VM image when they run
and, when restarted, they revert to the original VM image state.
A personal vDisk catalog is similar to a single-image catalog, but it also uses
personal vDisk technology.

- PowerManaged

  This catalog kind is for managed machines that are manually
  provisioned by administrators. All machines in this type of catalog
  are managed, and so must be associated with a hypervisor connection.

- Unmanaged

  This catalog kind is for unmanaged machines, so there is no
  associated hypervisor connection.

- PVS

  This catalog kind is for managed machines that are provisioned
  using Provisioning Services. All machines in this type of
  catalog are managed, and so must be associated with a hypervisor
  connection. Only shared desktops are suitable for this catalog kind.

Type:	CatalogKind
Position:	4
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningType

Specifies the ProvisioningType for the catalog. Values can be:

- Manual - No provisioning.
- PVS - Machine provisioned by PVS (machine may be physical, blade, VM,…).
- MCS - Machine provisioned by MCS (machine must be VM).

Type:	ProvisioningType
Position:	4

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SessionSupport

Specifies whether machines in the catalog are single or multi-session capable. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	5
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PersistUserChanges

Specifies how user changes are persisted on machines in the catalog. Possible values are:

- OnLocal: User changes are stored on the machine's local storage.
- Discard: User changes are discarded.

Type:	PersistUserChanges
Position:	6
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PvsForVM

Deprecated:

Identifies the provisioning scheme used by this catalog. To be specified in the format: ProvisioningSchemeGuid:ServiceGroupGuid. Applicable only to thin-cloned, single-image or personal vDisk catalogs.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AdminFolder

The folder in which the new catalog should reside (if any).

Type:	AdminFolder
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description for the catalog.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-IsRemotePC

Specifies whether this is to be a Remote PC catalog.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- MachinesArePhysical is true.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachinesArePhysical

Specifies whether machines in the catalog can be power-managed by the Citrix Broker Service. Where the Citrix Broker Service cannot control the power state of themachine specify $true, otherwise $false. Can only be specified together with a provisioning type of Pvs or Manual, or if used with the legacy CatalogKind parameter only with a Pvs catalog kind.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MdmEnrollment

MdmEnrollment for this catalog. Values can be:

- None - No enrollment.
- Intune - Microsoft Intune.

Type:	String
Accepted values:	Intune, IntuneWithCitrixTags, None
Position:	Named
Default value:	If no value is provided, the catalog does not enroll with Mdm.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MinimumFunctionalLevel

The minimum FunctionalLevel required for machines to register in the site.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25, L7_30, L7_34

Type:	FunctionalLevel
Position:	Named
Default value:	The default value for new Catalogs comes from the DefaultMinimumFunctionalLevel property of the Site object (see Get-BrokerSite).
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PvsAddress

Specifies the URL of the Provisioning Services server. Only applicable to Provisioning Services or Provisioning Services-personal vDisk catalogs.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PvsDomain

Specifies the Active Directory domain of the Provisioning Services server. Only applicable to Provisioning Services or Provisioning Services-personal vDisk catalogs.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RemotePCHypervisorConnectionUid

Specifies the hypervisor connection to use for powering on remote PCs in this catalog (only allowed when IsRemotePC is true).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Scope

Specifies the name of the delegated administration scope to which the catalog belongs.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with catalog. Must always be specified in multitenant sites, must not be specified otherwise.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TimeZone

The time zone in which this catalog's machines reside.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-UUID

An optional GUID for this catalog.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ZoneUid

Zone Uid associated with this catalog.

Type:	Guid
Position:	Named
Default value:	If no Uid is provided the catalog is associated with Primary Zone.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeId

Specifies the identity of the MCS provisioning scheme the new catalog is associated with (can only be specified for new catalogs with a ProvisioningType of MCS).

Type:	Guid
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Catalog

New-BrokerCatalog returns the created catalog.

Related Links

- about_Broker_RemotePC
- Get-BrokerCatalog
- Rename-BrokerCatalog
- Remove-BrokerCatalog
- Set-BrokerCatalog
- Move-BrokerCatalog

New-BrokerCatalogRebootSchedule

March 11, 2024

Creates a new reboot schedule for a catalog.

Syntax

```
New-BrokerCatalogRebootSchedule
   -CatalogUid <Int32>
   [-Description <String>]
   [-Enabled <Boolean>]
   [-MaxOvertimeStartMins <Int32>]
   [-Name] <String>
   -RebootDuration <Int32>
   [-StartDate <String>]
   [-StartTime <TimeSpan>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerCatalogRebootSchedule
   -CatalogName <String>
   [-Description <String>]
```



```
    [-Enabled <Boolean>]
    [-MaxOvertimeStartMins <Int32>]
    [-Name] <String>
    -RebootDuration <Int32>
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerCatalogRebootSchedule cmdlet is used to define a reboot schedule for a catalog.

Examples

EXAMPLE 1

Schedules the machines in the catalog named ‘BankTellers’to be rebooted Feb 3, 2021 between 2 AM and 4 AM.

```
New-BrokerCatalogRebootSchedule -Name BankTellers -CatalogName
    BankTellers  -StartDate "2022-02-03" -StartTime "02:00" -Enabled
    $true -RebootDuration 120
```

EXAMPLE 2

Schedules the machines in the catalog having Uid 17 to be rebooted between 1 AM and 5 AM at Feb 3, 2021. Ten minutes prior to rebooting, each machine will display a message box with the title “WARNING: Reboot pending”and message “Save your work”in every user session.

```
New-BrokerCatalogRebootSchedule -Name 'Update reboot' -CatalogUid 17 -
    StartDate "2022-02-03" -StartTime "01:00" -Enabled $true -
    RebootDuration 240 -WarningTitle "WARNING: Reboot pending" -
    WarningMessage "Save your work" -WarningDuration 10
```


Parameters

-Name

The name of the new reboot schedule.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CatalogName

The name of the catalog that this reboot schedule is applied to.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CatalogUid

The Uid of the catalog that this reboot schedule is applied to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

An optional description for the reboot schedule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Boolean that indicates if the new reboot schedule is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-StartDate

The date on which the schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.CatalogRebootSchedule

Related Links

- Get-BrokerCatalogRebootSchedule
- Set-BrokerCatalogRebootSchedule
- Remove-BrokerCatalogRebootSchedule
- Rename-BrokerCatalogRebootSchedule
- Start-BrokerRebootCycle

New-BrokerConfigurationSlot

March 11, 2024

Creates a new configuration slot.

Syntax

```
New-BrokerConfigurationSlot
   [-Description <String>]
   [-Name] <String>
   -SettingsGroup <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a new configuration slot. The SettingsGroup of the slot determines the particular collection of related settings that may be specified in a machine configuration associated with this slot.

For example, the configuration slot may be restricted to configuring Citrix User Profile Manager settings by specifying the SettingsGroup parameter as "G=UPM".

Examples

EXAMPLE 1

Create a new slot named "UPM"to configure settings specific to "User Profile Management"

```
New-BrokerConfigurationSlot -Name "UPM" -SettingsGroup "G=UPM"
```

Parameters

-Name

Name of the new configuration slot. This must be alphanumeric and not contain white space.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-SettingsGroup

The settings group determines the particular collection of related settings that may be controlled by this slot.

This must match the format of a Citrix Group Policy configuration group (e.g. "G=UPM").

Only settings that have this exact group may be specified in a machine configuration associated with this configuration slot.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Description of configuration slot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ConfigurationSlot

New-BrokerConfigurationSlot returns an object representing the newly created configuration slot

Related Links

- Get-BrokerConfigurationSlot
- Remove-BrokerConfigurationSlot
- New-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

New-BrokerConfiguredFTA

March 11, 2024

Creates a file type association with a published application.

Syntax

```
New-BrokerConfiguredFTA
   -ApplicationUid <Int32>
   -ImportedFTA <ImportedFTA>
   [-UUID <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerConfiguredFTA
   -ApplicationUid <Int32>
   -ExtensionName <String>
   [-ContentType <String>]
   -HandlerName <String>
   [-HandlerDescription <String>]
   [-HandlerOpenArguments <String>]
   [-UUID <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates an association between a file type and a published application for the purposes of the content redirection.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user

clicks on a document it launches the appropriate published application. This is known as "content redirection".

Configured file type associations are different from imported file type associations. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection. Imported file type associations are lists of known file type associations for a given desktop group. See Update-BrokerImportedFTA for more information about imported file type associations.

This cmdlet has two parameter sets, which correspond to the cmdlet's two use cases.

The first use case leverages imported file type associations to configure file types for published applications. Information about the file type association is read from the imported object. See the Update-BrokerImportedFTA cmdlet for more information about importing file type associations from a worker machine.

The second use case is more complex and allows you to create your own file type association without having to import it first. This also lets you create custom file type associations that may not already exist on the worker machines. This use case is more error-prone, however, because the individual attributes of the file type association must be correctly specified by you.

Examples

EXAMPLE 1

Gets the Uid for the application, gets the ImportedFTA

object for the file extension, and finally associates ".txt"with the

published "Notepad"application.

Note that the Get-BrokerImportedFTA cmdlet may return more than one

ImportedFTA objects for a specific extension name. See the help for

that cmdlet for more details.

```
$app = Get-BrokerApplication "Notepad"
$fta = Get-BrokerImportedFTA -ExtensionName ".txt"
New-BrokerConfiguredFTA -ImportedFTA $fta -ApplicationUid $app.Uid
```

EXAMPLE 2

This example is identical to the first, but shows the

the second use case of the cmdlet, specifying each attribute manually.


```
$app = Get-BrokerApplication "Notepad"
New-BrokerConfiguredFTA -ApplicationUid $app.Uid -ExtensionName ".txt"
    -HandlerName "txtfile" -ContentType "text\plain" -HandlerDescription
     "Text Document" -HandlerOpenArguments "%1"
```

Parameters

-ApplicationUid

Specifies the application with which the file type should be associated.

Type:	Int32
Position:	Named
Default value:	(required)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ImportedFTA

Specifies the ImportedFTA object to use for creating the ConfiguredFTA object. All values needed to create a ConfiguredFTA object are read from the ImportedFTA object.

Type:	ImportedFTA
Position:	Named
Default value:	(required)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExtensionName

Specifies the extension name for the file type association.

For example, “.txt”or “.doc”.

Type:	String
Position:	Named
Default value:	(required)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HandlerName

Specifies the name of the handler for the file type association (as seen in the Registry). For example, “TXTFILE”or “Word.Document.8”.

Type:	String
Position:	Named
Default value:	(required)
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-UUID

An optional GUID for this ConfiguredFTA.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ContentType

Specifies the content type of the file type (as listed in
the Registry). For example, content type would be “text/plain”or
“application/msword”.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HandlerDescription

Specifies the description of the handler for the file type

association.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HandlerOpenArguments

Specifies the arguments for the open command that the handler should use. For example, "%1".

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.ConfiguredFTA

This cmdlet returns a single ConfiguredFTA object.

Related Links

- Get-BrokerImportedFTA
- Get-BrokerConfiguredFTA
- Remove-BrokerConfiguredFTA

New-BrokerDelayedHostingPowerAction

March 11, 2024

Causes a power action to be queued after a delay.

Syntax

```
New-BrokerDelayedHostingPowerAction
   -Action <PowerManagementAction>
   -Delay <TimeSpan>
   [-MachineName] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Causes a power action to be queued after the specified period of time.

Only ShutDown or Suspend actions can be requested to be delayed in this manner.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement' .

Examples

EXAMPLE 1

Causes the machine called XD_VDA1 to be shut down after a delay of two minutes.

```
New-BrokerDelayedHostingPowerAction -Action Shutdown -MachineName '
    XD_VDA1' -Delay '00:02:00'
```

Parameters

-MachineName

Specifies the machine that the action is to be performed on.

The machine can be identified by DNS name, short name, SID, or name of the form domain\machine.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Action

Specifies the power state change action that is to be performed on the specified machine after the specified delay.

Valid values are Shutdown and Suspend.

Type:	PowerManagementAction
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Delay

Specifies a timespan delay before the action is queued.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DelayedHostingPowerAction

New-BrokerDelayedHostingPowerAction returns the created delayed power action.

Related Links

- about_Broker_PowerManagement
- Get-BrokerDelayedHostingPowerAction
- New-BrokerDelayedHostingPowerAction

New-BrokerDesktopGroup

March 11, 2024

Create a new desktop group for managing the brokering of groups of desktops.

Syntax

```
New-BrokerDesktopGroup
   [-AdminFolder <AdminFolder>]
   [-AllowReconnectInMaintenanceMode <Boolean>]
   [-AppDisks <Guid[]>]
   [-AppProtectionKeyLoggingRequired <Boolean>]
   [-AppProtectionScreenCaptureRequired <Boolean>]
   [-AutomaticPowerOnForAssigned <Boolean>]
```



```
    [-AutomaticPowerOnForAssignedDuringPeak <Boolean>]
    [-AutomaticRestartForUntaggedMachines <Boolean>]
    [-AutoscaleLogOffReminderEnabled <Boolean>]
    [-AutoscaleLogOffReminderIntervalSecondsOffPeak <Int32>]
    [-AutoscaleLogOffReminderIntervalSecondsPeak <Int32>]
    [-AutoscaleLogOffReminderMessage <String>]
    [-AutoscaleLogOffReminderTitle <String>]
    [-AutoscaleLogOffWarningMessage <String>]
    [-AutoscaleLogOffWarningTitle <String>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak <Int32>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak <Int32>]
    [-AutoscalingEnabled <Boolean>]
    [-ColorDepth <ColorDepth>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    -DesktopKind <DesktopKind>
    [-DisconnectOffPeakIdleSessionAfterSeconds <Int32>]
    [-DisconnectPeakIdleSessionAfterSeconds <Int32>]
    [-Enabled <Boolean>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IsRemotePC <Boolean>]
    [-LicenseModel <LicenseModel>]
    [-LogoffOffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-LogoffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-MachineCost <Decimal>]
    [-MachineLogOnType <MachineLogOnType>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-Name] <String>
    [-OffPeakBufferSizePercent <Int32>]
    [-OffPeakDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakDisconnectTimeout <Int32>]
    [-OffPeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakExtendedDisconnectTimeout <Int32>]
    [-OffPeakLogOffAction <SessionChangeHostingAction>]
    [-OffPeakLogOffTimeout <Int32>]
    [-PeakAutoscaleAssignedPowerOnIdleAction <String>]
    [-PeakAutoscaleAssignedPowerOnIdleTimeout <Int32>]
    [-PeakBufferSizePercent <Int32>]
    [-PeakDisconnectAction <SessionChangeHostingAction>]
    [-PeakDisconnectTimeout <Int32>]
    [-PeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-PeakExtendedDisconnectTimeout <Int32>]
    [-PeakLogOffAction <SessionChangeHostingAction>]
    [-PeakLogOffTimeout <Int32>]
    [-PolicySetGuid <Guid>]
    [-PowerOffDelay <Int32>]
    [-ProductCode <String>]
    [-ProtocolPriority <String[]>]
    [-PublishedName <String>]
    [-RequiredSleepCapability <String>]
    [-ResourceLeasingEnabled <Boolean>]
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak <Int32>]
```



```
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak <Int32>]
    [-RestrictAutoscaleTagUid <Int32>]
    [-ReuseMachinesWithoutShutdownInOutage <Boolean>]
    [-Scope <String[]>]
    [-SecureIcaRequired <Boolean>]
    [-SessionSupport <SessionSupport>]
    [-SettlementPeriodBeforeAutoShutdown <TimeSpan>]
    [-SettlementPeriodBeforeUse <TimeSpan>]
    [-ShutdownDesktopsAfterUse <Boolean>]
    [-TenantId <Guid>]
    [-TimeZone <String>]
    [-TurnOnAddedMachine <Boolean>]
    [-UseVerticalScaling <Boolean>]
    [-UUID <Guid>]
    [-ZonePreferences <ZonePreference[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerDesktopGroup cmdlet creates a new broker desktop group that can then be used to manage the brokering settings of all desktops within that desktop group. Once the desktop group has been created, you can create desktops in it by adding the appropriate broker machines to it using the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets.

Desktop groups hold settings that apply to all desktops they contain.

For any automatic power management settings of a desktop group to take effect, the group's TimeZone property must be specified. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.

Examples

EXAMPLE 1

Create a desktop group to manage the brokering of private desktops, which will appear to users with the name "MyDesktop".

```
New-BrokerDesktopGroup "Assigned Desktops" -PublishedName "MyDesktop" -
    DesktopKind Private
```


Parameters

-Name

The name of the new broker desktop group.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopKind

The kind of desktops this group will hold. Valid values are Private and Shared.

Type:	DesktopKind
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AdminFolder

The folder in which the new desktop group should reside (if any).

Type:	AdminFolder
Position:	Named
Default value:	None (root folder)
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-AllowReconnectInMaintenanceMode

Whether the maintenance mode allows session reconnect to various Session support types.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppDisks

Specifies the application disks to be used by machines in the desktop group.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppProtectionKeyLoggingRequired

Specifies whether key logging app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AppProtectionScreenCaptureRequired

Specifies whether screen capture app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutomaticPowerOnForAssigned

Specifies whether assigned desktops in the desktop group should be automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutomaticPowerOnForAssignedDuringPeak

Specifies whether assigned desktops in the desktop group should be automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutomaticRestartForUntaggedMachines

Indicates whether untagged single-session machines belonging to a desktop group configured for restrict Autoscale and shutdown after use should restart after untainting.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscaleLogOffReminderEnabled

Boolean value indicating whether the warning messages should be sent on an interval to nudge a logoff should be sent on an interval when autoscale is enabled.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsOffPeak

Represents the interval in seconds for which messages are sent to the user during off peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsPeak

Represents the interval in seconds for which messages are sent to the user during peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscaleLogOffReminderMessage

Specifies the notification message to display to users in active sessions belonging to machines needed by Autoscale for shutdown

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AutoscaleLogOffReminderTitle

Specifies the notification message dialog title displayed when Autoscale issues a logoff reminder request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AutoscaleLogOffWarningMessage

Specifies the warning message to display to users in active sessions prior to Autoscale issuing a logoff request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AutoscaleLogOffWarningTitle

Specifies the warning message dialog title displayed prior to Autoscale issuing a logoff request.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak

Specifies the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group during off-peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak

Specifies the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group during peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AutoscalingEnabled

Specifies whether machines in this desktop group can be Autoscaled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ColorDepth

Specifies the color depth that the ICA session should use for desktops in this group. Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	TwentyFourBit
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DeliveryType

Specifies whether desktops, applications, or both, can be delivered from machines contained within the new desktop group. Desktop groups with a DesktopKind of Private cannot be used to deliver both desktops and applications. Defaults to DesktopsOnly if not specified.

Valid values are DesktopsOnly, AppsOnly, and DesktopsAndApps.

Type:	DeliveryType
Position:	Named
Default value:	DesktopsOnly

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A description for this desktop group useful for administrators of the site.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DisconnectOffPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during off-peak time.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DisconnectPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during peak time.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Whether the desktop group should be in the enabled state; disabled desktop groups do not appear to users.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IconUid

The UID of the broker icon to be displayed to users for their desktop(s) in this desktop group.

Type:	Int32
Position:	Named
Default value:	The Uid of the default desktop icon in this site - use the Get-BrokerSite cmdlet to find this value.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-InMaintenanceMode

Whether the desktop should be created in maintenance mode; a desktop group in maintenance mode will not allow users to connect or reconnect to their desktops.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IsRemotePC

Specifies whether this is to be a Remote PC desktop group.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- DeliveryType is DesktopsOnly
- DesktopKind is Private

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LicenseModel

The license model for this desktop group. If none is specified, then the site-wide license model is used.

Type:	LicenseModel

Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LogoffOffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during off-peak time.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LogoffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during peak time.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachineCost

The per-hour instance cost of machines in this desktop group.

Type:	Decimal
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachineLogOnType

Specifies the login type for the desktop group. Values can be:

- ActiveDirectory - Perform Active Directory login on the machine.
- LocalMappedAccount - Perform local mapped account login on the machine.

Type:	MachineLogOnType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MinimumFunctionalLevel

The minimum FunctionalLevel required for machines to work successfully in the desktop group.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25

Type:	FunctionalLevel
Position:	Named
Default value:	The FunctionalLevel of the current release (L7_6); by default no machines with less than the most current FunctionalLevel will be functional.
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-OffPeakBufferSizePercent

The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available outside peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakDisconnectAction

The action to be performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakDisconnectTimeout

The number of minutes before the configured action should be performed after a user session disconnects outside peak hours.

Type:	Int32

Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakExtendedDisconnectAction

The action to be performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakExtendedDisconnectTimeout

The number of minutes before the second configured action should be performed after a user session disconnects outside peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakLogOffAction

The action to be performed after a configurable period of a user session ending outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OffPeakLogOffTimeout

The number of minutes before the configured action should be performed after a user session ends outside peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleAction

The action to be performed on an assigned machine started by Autoscale if that machine then remains unused for a defined period of time.

Type:	String
Accepted values:	Nothing, Shutdown, Suspend
Position:	Named
Default value:	Nothing

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleTimeout

The number of minutes before the configured action is performed on an assigned machine previously started by autoscale that subsequently remains unused.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakBufferSizePercent

The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available in peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakDisconnectAction

The action to be performed after a configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakDisconnectTimeout

The number of minutes before the configured action should be performed after a user session disconnects in peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakExtendedDisconnectAction

The action to be performed after a second configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakExtendedDisconnectTimeout

The number of minutes before the second configured action should be performed after a user session disconnects in peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakLogOffAction

The action to be performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	Nothing
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakLogOffTimeout

The number of minutes before the configured action should be performed after a user session ends in peak hours.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PolicySetGuid

The policy set assigned to this delivery group.

Type:	Guid
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PowerOffDelay

The number of minutes following a power-on operation that Auto Scale will wait before attempting to power off that same machine. Possible values in the range 0 - 60.

Type:	Int32
Position:	Named
Default value:	30
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProductCode

The licensing product code for this desktop group. If none is specified, then the site-wide product code is used.

Type:	String

Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProtocolPriority

A list of protocol names in the order in which they should be attempted for use during connection.

Type:	String[]
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PublishedName

The name that will be displayed to users for their desktop(s) in this desktop group.

Type:	String
Position:	Named
Default value:	The same value as that supplied for the name of the desktop group.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RequiredSleepCapability

The sleep capability of this desktop group. Possible values are None, or Suspend.

Type:	String
Accepted values:	None, Suspend
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ResourceLeasingEnabled

Indicates if this desktop group is enabled for resource leasing.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak

This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages 'tagged' machines, as per policy, in off-peak. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down 'tagged' machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for desktop groups configured for Restrict Autoscaling.

Type:	Int32
Position:	Named
Default value:	-1

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak

This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages 'tagged' machines, as per policy, in peak-time. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down 'tagged' machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for delivery groups configured for Restrict Autoscaling.

Type:	Int32
Position:	Named
Default value:	-1
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RestrictAutoscaleTagUid

If set to a Tag UID, only machines that have this tag will be auto scaled.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ReuseMachinesWithoutShutdownInOutage

Specifies whether power cycle operations are required during outage for pooled machines.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Scope

Specifies the name of the delegated administration scope to which the desktop group should belong.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SecureIcaRequired

Whether HDX connections to desktops in the new desktop group require the use of a secure protocol.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-SessionSupport

Specifies whether machines in the desktop group are single or multi-session capable. Values can be:

- SingleSession - Single-session only machine.
- MultiSession - Multi-session capable machine.

Type:	SessionSupport
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SettlementPeriodBeforeAutoShutdown

Time after a session ends during which automatic shutdown requests (for example, shutdown after use, idle pool management) are deferred. Any outstanding shutdown request takes effect after the settlement period expires. This is typically used to configure time to allow logoff scripts to complete.

Type:	TimeSpan
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SettlementPeriodBeforeUse

Idle period before a machine can be selected to host a new session after registration or the end of a previous session. This is typically used to allow a machine to become idle following processing

associated with start-up or logoff actions. A machine may still be selected during the idle period if no other machine is available for immediate use.

Type:	TimeSpan
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ShutdownDesktopsAfterUse

Whether desktops in this desktop group should be automatically shut down when each user session completes (only relevant to power-managed desktops).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TenantId

Specifies identity of tenant associated with desktop group. Must always be specified in multitenant sites, must not be specified otherwise.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TimeZone

The time zone in which this desktop group's machines reside.

The time zone must be specified for any of the group's automatic power management settings to take effect. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-TurnOnAddedMachine

This flag specifies whether the Broker Service should attempt to power on machines when they are added to the desktop group.

Type:	Boolean
Position:	Named
Default value:	$false for single session machines and $true for multi-session machines.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UseVerticalScaling

Determines whether to use vertical scaling when considering RDS machines in the desktop group for launches. Vertical scaling would saturate machines in the current pool rather than send sessions to the least loaded machines. This would be a trade in performance vs. cost, where vertical scaling would be less costly.

With the default value ($null), the site-wide value will be inherted. A value of $false indicates that horizontal scaling should be used.

Type:	Boolean
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

An optional GUID for this desktop group.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ZonePreferences

Ordered list of zone preferences to be applied when launching resources from this desktop group. Valid zone preference values are UserLocation, UserHome, UserHomeOnly and ApplicationHome.

The list can have zero or more entries subject to the following restrictions: Zone preferences can only be applied to groups having a DesktopKind of Shared; the same zone preference value cannot occur in the list more than once; the UserHome and UserHomeOnly values are mutually exclusive and cannot both appear in the list.

Type:	ZonePreference[]
Position:	Named
Default value:	For shared groups the default preference list is ApplicationHome, UserHome, then UserLocation.
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.DesktopGroup

The newly created desktop group.

Notes

Once a new desktop group is created, you can create desktops in it by adding the appropriate broker machines to it using the Add-BrokerMachine or Add-BrokerMachinesToDesktopGroup cmdlets.

Related Links

- about_Broker_Desktops
- about_Broker_PowerManagement
- about_Broker_RemotePC
- Get-BrokerDesktopGroup
- Set-BrokerDesktopGroup
- Rename-BrokerDesktopGroup
- Move-BrokerDesktopGroup
- Remove-BrokerDesktopGroup
- Add-BrokerMachine
- Add-BrokerMachinesToDesktopGroup
- Get-BrokerSite

New-BrokerDesktopGroupWebhook

March 11, 2024

Create a webhook for the sepcified desktop group

Syntax

```
New-BrokerDesktopGroupWebhook
   -Address <String>
   -DesktopGroupUid <Int32>
   -OnEvent <WebhookTrigger>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
7     [<CommonParameters>]
```

Description

This cmdlet is used to create a webhook for the sepcified desktop group, that is invoked upon the specified event of the desktop group.

Examples

EXAMPLE 1

Creates a new webhook for the desktop group with Uid 1, that is to be invoked when a machine registers to the broker.

When it is invoked, a HTTP POST request is sent out to the address above with a JSON payload {“SamName”: “value”} where “value”is the SAM name of registering machine.

```
1  New-BrokerDesktopGroupWebhook -DesktopGroupUid 1 -OnEvent
       MachineRegistration -Address 'http://citrix.com/example'
```

Parameters

-Address

The URL of the webhook

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

The Uid of the desktop group to configure a webhook

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OnEvent

The event upon that the webhook is invoked, currently the only supported event is MachineRegistration.

For MachineRegistration, when the webhook is invoked it is HTTP POST with a JSON payload of the format: {“SamName”: “value”} where the “value”is the SamName of the machine that is registering to the broker.

Type:	WebhookTrigger
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroupWebhook

The set of webhooks to be added to the desktop group can be piped into this cmdlet

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerDesktopGroup

New-BrokerEntitlementPolicyRule

March 11, 2024

Creates a new desktop rule in the site's entitlement policy.

Syntax

```
New-BrokerEntitlementPolicyRule
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    -DesktopGroupUid <Int32>
    [-Enabled <Boolean>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-IconUid <Int32>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-LeasingBehavior <LeasingBehavior>]
    [-MaxPerEntitlementInstances <Int32>]
    [-Name] <String>
    [-PublishedName <String>]
    [-RestrictToTag <String>]
    [-SecureIcaRequired <Boolean>]
    [-SessionReconnection <SessionReconnection>]
    [-UUID <Guid>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerEntitlementPolicyRule cmdlet adds a new desktop rule to the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

The following constraints apply when creating a desktop entitlement rule for a desktop group:

- The group's desktop kind must be Shared
- The group's delivery type must be DesktopsOnly or DesktopsAndApps

When a user selects a desktop entitlement published from a shared group, a machine is selected from the group on which to run the desktop session. No permanent association exists between the user and the selected machine; once the session ends the association also ends.

Multiple desktop rules in the entitlement policy can apply to the same desktop group. Where a user is granted an entitlement by more than one rule for the same group, they can use as many desktop sessions at the same time as they have entitlements.

Examples

EXAMPLE 1

Creates an desktop rule in the entitlement policy that entitles all members of the SUPPORT\uk-staff group to a desktop session from the Customer Support desktop group. The desktop entitlement name seen by users is Support Desktop.

```
$dg = Get-BrokerDesktopGroup 'Customer Support'
New-BrokerEntitlementPolicyRule 'UK Office' -DesktopGroupUid $dg.Uid -
    IncludedUsers support\uk-staff -PublishedName 'Support Desktop'
```

Parameters

-Name

Specifies the administrative name of the new desktop rule. Each rule in the site's entitlement policy must have a unique name (irrespective of whether they are desktop or application rules).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

Specifies the unique ID of the desktop group to which the new desktop rule applies.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ColorDepth

Specifies the color depth of any desktop sessions launched by a user from this entitlement.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	ColorDepth
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies an optional description of the new desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new desktop rule is initially enabled. A disabled rule is ignored when evaluating the site's entitlement policy.

Type:	Boolean

Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Specifies whether the excluded users filter is initially enabled. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExcludedUsers

Specifies the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from the new rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IconUid

Specifies the unique ID of the icon used to display the desktop session entitlement to the user.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Int32
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Specifies whether the included users filter is initially enabled. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a desktop session by the new rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IncludedUsers

Specifies the included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the new rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	(empty list)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LeasingBehavior

Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:

Allowed and Disallowed.

The Allowed value indicates that connection leasing should behave normally. The Disallowed value prevents users

from launching or reconnecting to sessions using this entitlement while connection leasing is active (typically during a database outage).

Type:	LeasingBehavior
Position:	Named
Default value:	Allowed
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxPerEntitlementInstances

Maximum allowed concurrently running instances of the desktop associated with this entitlement in the site . A value of zero allows unlimited usage.

Type:	Int32
Position:	Named
Default value:	0

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PublishedName

The name of the new desktop session entitlement as seen by the user.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RestrictToTag

Optional tag that may be used further to restrict which machines may be made accessible to a user by an entitlement policy rule. A machine may be made accessible by an entitlement policy rule only if either the rule has no tag restriction or the rule does have a tag restriction and the machine is tagged with the same tag.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-SecureIcaRequired

Specifies whether the new desktop rule requires the SecureICA protocol for desktop sessions launched using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Boolean
Position:	Named
Default value:	Null (dynamically inherited from the desktop group)
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SessionReconnection

Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	Always
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

An optional GUID for this rule.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

New-BrokerEntitlementPolicyRule returns the newly created desktop rule in the entitlement policy.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_ErrorHandling
- Get-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule

New-BrokerGpoFilter

March 11, 2024

Creates a new GPO filter.

Syntax

```
New-BrokerGpoFilter
   [-FilterData <String>]
   -FilterType <String>
   -IsAllowed <Boolean>
   -IsEnabled <Boolean>
   -PolicyGuid <Guid>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a GPO filter for a policy to specify how the policy should be applied.

Examples

EXAMPLE 1

Creates a new client name filter for a policy.

```
New-BrokerGpoFilter -PolicyGuid ([Guid]"12345678-...") -FilterType
    ClientName -FilterData 'Client1' -IsAllowed $true -IsEnabled $true
```

Parameters

-FilterType

The filter type.

Type:	String
Accepted values:	AccessControl, BranchRepeater, ClientIP, ClientName, DesktopGroup, DesktopKind, DesktopTag, OU, User
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IsAllowed

Specifies to allow the policy when the filter condition is true.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IsEnabled

Specifies if the filter is enabled or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PolicyGuid

The GUID of the policy for which the filter is defined

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-FilterData

The filter data. Ignored if the filter is a BranchRepeater filter.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoFilter

Outputs the GPO filter object.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoFilter
- Set-BrokerGpoFilter
- Remove-BrokerGpoFilter

New-BrokerGpoPolicy

March 11, 2024

Create a new GPO policy.

Syntax

```
New-BrokerGpoPolicy
   [-Description <String>]
   [-IsEnabled <Boolean>]
   [-Name] <String>
   -PolicySetGuid <Guid>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerGpoPolicy cmdlet creates a new GPO policy.

Examples

EXAMPLE 1

Create a policy named Policy0 in the default site policy set.

```
New-BrokerGpoPolicy -Name "Policy0"
```


Parameters

-Name

Specifies the new policy name. The name must not exist in the specified policy set.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PolicySetGuid

Specifies the policy set the new policy will be in. If the GUID is not specified, the new policy is created in the default site policy set.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies the description of the policy.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-IsEnabled

Specifies whether policy is enabled.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicy

The newly created GPO policy.

Notes

The priority of the new policy is assigned with the lowest priority.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoPolicy
- Set-BrokerGpoPolicy
- Remove-BrokerGpoPolicy

New-BrokerGpoPolicySet

March 11, 2024

Creates a new GPO policy set.

Syntax

```
New-BrokerGpoPolicySet
    [-Description <String>]
    [-Name] <String>
    -PolicySetType <String>
    [-Scope <String[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Create a new GPO policy set.

Examples

EXAMPLE 1

Creates a new policy set for a configuration slot without a description. The caller should keep the PolicySetGuid property returned in the object for later use.

```
New-BrokerGpoPolicySet -PolicySetType SlotPolicies
```

Parameters

-Name

The name of the policy set. It must be unique among all policy sets.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PolicySetType

The type of the policy set. Valid values are: SitePolicies, SlotPolicies, and SiteTemplate. SitePolicies specifies a policy set that stores the default policies for the site. SlotPolicies specifies a policy set that stores the policies for a configuration slot. SiteTemplate specifies a policy set that stores the site's policy templates.

Type:	String
Accepted values:	CustomTemplates, DeliveryGroupPolicies, SitePolicies, SiteTemplates
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

A short description describing the use of this policy set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Scope

Specifies the name of the delegated administration scope to which the policy set belongs.

Type:	String[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicySet

New-BrokerGpoPolicySet returns a GPO policy set object.

Related Links

- about_Broker_GroupPolicy

New-BrokerGpoSetting

March 11, 2024

Creates a new GPO setting.

Syntax

```
New-BrokerGpoSetting
   -PolicyGuid <Guid>
   [-SettingName] <String>
   [-SettingValue <String>]
   [-UseDefault <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a GPO setting for a policy.

Examples

EXAMPLE 1

Creates a new 'Audio quality'setting.

```
New-BrokerGpoSetting -PolicyGuid ([Guid]"12345678-...") -SettingName
    AudioQuality -UseDefault $true
```


Parameters

-SettingName

The setting name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PolicyGuid

The GUID of the policy for which the filter is created.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SettingValue

The setting value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-UseDefault

Indicate if the default value of setting is used. Ignored if setting is Boolean.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoSetting

Outputs the GPO setting object.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoSetting
- Set-BrokerGpoSetting
- Remove-BrokerGpoSetting

New-BrokerHostingPowerAction

March 11, 2024

Creates a new action in the power action queue.

Syntax

```
New-BrokerHostingPowerAction
   -Action <PowerManagementAction>
   [-ActualPriority <Int32>]
   [-MachineName] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
7     [<CommonParameters>]
```

Description

The New-BrokerHostingPowerAction cmdlet adds a new power action record into the queue of power actions to be performed. The power actions in the queue are processed on a priority basis and sent to the relevant hypervisor to change the power state of a virtual machine.

A power action record defines the action to be performed, the machine on which the action is to be performed, and an initial priority value for the action. Multiple actions may be created that relate to the same machine.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement' .

Examples

EXAMPLE 1

Causes the machine called 'XD_VDA1'to be shut down.

```
1  New-BrokerHostingPowerAction -Action Shutdown -MachineName 'XD_VDA1'
```

Parameters

-MachineName

Specifies the machine that the action is to be performed on.

The machine can be identified by DNS name or short name or SID or 'machine\domain'form name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Action

Specifies the power state change action that is to be performed on the specified machine.

Valid values are: TurnOn, TurnOff, ShutDown, Reset, Restart, Suspend and Resume.

Type:	PowerManagementAction
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ActualPriority

Specifies an initial priority value for the action in the queue.

This priority is the current action priority; the 'base'priority for actions created via this cmdlet is always 30. Numerically lower priority values indicate more important actions that are processed in preference to actions with numerically higher priority settings.

Type:	Int32
Position:	Named
Default value:	30
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

MachineName (System.String)

A list of machine names can be supplied as input.

Outputs

Citrix.Broker.Admin.SDK.HostingPowerAction

New-BrokerHostingPowerAction returns the newly created power action record.

Related Links

- about_Broker_PowerManagement
- Get-BrokerHostingPowerAction
- Set-BrokerHostingPowerAction
- Remove-BrokerHostingPowerAction

New-BrokerHypervisorConnection

March 11, 2024

Creates a new hypervisor connection.

Syntax

```
New-BrokerHypervisorConnection
   [-HypHypervisorConnectionUid] <Guid>
   [-PreferredController <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerHypervisorConnection cmdlet creates a new hypervisor connection.

Examples

EXAMPLE 1

This command creates a new hypervisor connection with a preferred controller.

```
New-BrokerHypervisorConnection -PreferredController "domainName\
   controllerName" -HypHypervisorConnectionUid "d16f4e56-b85e-4ba6-b745
   -0e978ae4f192"
```

EXAMPLE 2

This command creates a new hypervisor connection, and leaves it to the system to select a preferred controller.

```
New-BrokerHypervisorConnection -HypHypervisorConnectionUid "d16f4e56-
   b85e-4ba6-b745-0e978ae4f192"
```


Parameters

-HypHypervisorConnectionUid

The Guid that identifies the hypervisor connection, as defined in DUM.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreferredController

The preferred controller machine for the hypervisor connection. Can be specified as (first match is used):

- Full SAM name.
- Full DNS name.
- SID value.
- NetBIOS name (SAM without domain).
- Partial DNS name (DNS name without some or all domain information).

Where not specified, the system selects preferred controller machine based on loading.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.HypervisorConnection

New-BrokerHypervisorConnection returns an opaque object containing information about the hypervisor connection.

Related Links

- Get-BrokerHypervisorConnection
- Remove-BrokerHypervisorConnection
- Set-BrokerHypervisorConnection

New-BrokerIcon

March 11, 2024

Creates a new icon.

Syntax

```
New-BrokerIcon
   [-EncodedIconData] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Accepts Base64 encoded .ICO format icon data, stores it in the database and returns an Icon object containing the Uid assigned to it.

New-BrokerIcon can be used with the Get-BrokerIcon cmdlet to obtain the Base64 icon. See Examples for a demonstration.

Examples

EXAMPLE 1

Extracts the first icon resource from notepad.exe, and sets this as the icon for a desktop group.

```
$ctxIcon = Get-BrokerIcon -FileName C:\Windows\System32\notepad.exe -
   index 0
$brokerIcon = New-BrokerIcon -EncodedIconData $ctxIcon.EncodedIconData
$desktopGroup = Get-BrokerDesktopGroup -Name 'MyDesktopGroup'
Set-BrokerDesktopGroup $desktopGroup -IconUid $brokerIcon.Uid
```


Parameters

-EncodedIconData

Specifies the Base64 encoded .ICO format icon data.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Icon

Returns an Icon object.

Related Links

- Get-BrokerIcon
- Remove-BrokerIcon

New-BrokerImportDb

March 11, 2024

This cmdlet is for internal use only

Syntax

```
New-BrokerImportDb
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

String

Related Links

New-BrokerLocalDb

March 11, 2024

This cmdlet is for internal use only

Syntax

```
New-BrokerLocalDb
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

String

Related Links

New-BrokerMachine

March 11, 2024

Adds a machine that can be used to run desktops and applications.

Syntax

```
New-BrokerMachine
   [-AssignedClientName <String>]
   [-AssignedIPAddress <String>]
   -CatalogUid <Int32>
   [-HostedMachineId <String>]
   [-HypervisorConnectionUid <Int32>]
   [-InMaintenanceMode <Boolean>]
   [-IsReserved <Boolean>]
```



```
    [-MachineName] <String>
    [-UUID <Guid>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

By adding a machine to a catalog, New-BrokerMachine adds a machine to the site, and is the first step in making the machine available to run users'desktops and applications. The machine may be physical or virtual.

For physical machines, you must specify the machine's SID and the catalog to which it will belong. For virtual machines which are not provisioned by MCS, you must also provide the hypervisor connection responsible for running the machine and the hosted machine ID by which the hypervisor recognizes the machine.

The machine must support the expected capabilities of the catalog: the catalog specifies a SessionType and a MinimalFunctionalLevel. The session support of the machine is determined by the type of Citrix VDA software installed (server or workstation) and the functional level depends on the version of the Citrix VDA software installed. The New-BrokerMachine command will complete successfully if these are not correct but the machine will be unable to register.

For more information about machines, see about_Broker_Machines.

Examples

EXAMPLE 1

This adds the physical machine with the specified SAM name to this site and places it in the specified catalog.

```
New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine'
```

EXAMPLE 2

This adds the physical machine with the specified SID to this site and places it in the specified catalog.

```
New-BrokerMachine -CatalogUid 2 -MachineName 'S
    -1-5-12-1234567890-1234567890-1234567890-1234'
```


EXAMPLE 3

This adds the virtual machine, running on the specified hypervisor, to this site and places it in the catalog.

```
New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine' -
    HostedMachineId 'F8143B4F-7371-4efa-868A-54787EF9F64E' -
    HypervisorConnectionUid 5
```

EXAMPLE 4

This adds the specified physical machine to the site and uses Add-BrokerMachine to add it to a desktop group.

```
$m = New-BrokerMachine -CatalogUid 2 -MachineName 'domain\machine'
Add-BrokerMachine -InputObject $m -DesktopGroup 3
```

Parameters

-MachineName

Specify the name of the machine to create (in the form 'domain\machine'). A SID can also be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CatalogUid

The catalog to which this machine will belong.

Type:	Int32
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-AssignedClientName

The client name to which this machine will be assigned. Machines can be assigned to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AssignedIPAddress

The client IP address to which this machine will be assigned. Machines can be assigned to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HostedMachineId

The unique ID by which the hypervisor recognizes the machine. Omit this for physical machines or MCS-provisioned VMs.

Type:	String
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HypervisorConnectionUid

The hypervisor connection that runs the machine. Omit this for physical machines or MCS-provisioned VMs.

Type:	Int32
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-InMaintenanceMode

Specifies whether the machine is initially in maintenance mode. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IsReserved

Specifies whether the machine should be reserved for special use, for example, for AppDisk preparation. A reserved machine cannot be added to a desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

An optional GUID for this machine.

Type:	Guid
Position:	Named
Default value:	A new GUID is generated if none is supplied.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Machine

New-BrokerMachine returns the created machine.

Related Links

- about_Broker_Machines
- Add-BrokerMachine

New-BrokerMachineCommand

March 11, 2024

Creates a new command to deliver to a desktop.

Syntax

```
New-BrokerMachineCommand
   -Category <String>
   -CommandName <String>
   [-CommandData <Byte[]>]
   [-DesktopGroups <DesktopGroup[]>]
   [-SendTrigger <MachineCommandTrigger>]
   -User <String>
   [-SendDeadline <TimeSpan>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerMachineCommand
   -Category <String>
   -CommandName <String>
   [-CommandData <Byte[]>]
   [-SendTrigger <MachineCommandTrigger>]
   -MachineUid <Int32>
   [-SendDeadline <TimeSpan>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerMachineCommand
   -Category <String>
   -CommandName <String>
   [-CommandData <Byte[]>]
   [-SendTrigger <MachineCommandTrigger>]
   -SessionUid <Int64>
   [-SendDeadline <TimeSpan>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerMachineCommand
   -Category <String>
   -CommandName <String>
   [-CommandData <Byte[]>]
   [-Synchronous]
   -MachineUid <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Create a new command queued for delivery to a desktop. Commands are sent to a specific handler installed on the desktop using the Category parameter. Each handler has its own list of commands

identified by the CommandName parameter. Optional command data can be provided using the CommandData parameter in a format specified by the handler.

Commands are targeted at a specific user, session or machine. Commands targeted at a user can be further be restricted to one or more desktop groups.

The SendTrigger is used to restrict the command to a specific event related to the target. For example, when the target machine registers or when the target user reconnects to a session. The command will be sent to the machine when the SendTrigger occurs for the target.

If the Synchronous switch is provided, the target must be a machine and no SendTrigger can be specified. The command is sent immediately to the machine if it is currently registered and fails if the machine is not registered.

Note that the combined length of the Category and CommandName is limited to 64 characters. The Category and CommandName must both be entirely alphanumeric and not include any white space.

Examples

EXAMPLE 1

Instruct the User Profile Manager service to execute the "ResetUpmProfile"command using the encoded $byteArray command data when user domain1\user1 logs on to any machine in desktop group 1

```
New-BrokerMachineCommand -Category UserProfileManager -CommandName "
    ResetUpmProfile" -DesktopGroups 1 -CommandData $byteArray -
    SendTrigger logon -user domain1\user1
```

EXAMPLE 2

Instruct the monitor service to immediately execute the "EnableLogging"command on the machine having Uid 13.

```
New-BrokerMachineCommand -Synchronous -Category "MonitorService" -
    CommandName "EnableLogging" -MachineUid 13
```

Parameters

-Category

The service on the desktop to send the command to.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CommandName

The name of the command to send (as defined by the service).

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-User

User whose desktop or session should receive the command.

Type:	String
Position:	Named
Default value:	Any user.
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-MachineUid

Specific machine that should receive the command.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SessionUid

Currently logged on user session that should receive the command.

Type:	Int64
Position:	Named
Default value:	Any session.
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Synchronous

Send the command immediately and block while waiting for the reply.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-CommandData

Optional additional data to include with the command.

Type:	Byte[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroups

Further restrict the command targeted at a user to machines in these desktop groups.

Type:	DesktopGroup[]
Position:	Named
Default value:	No restriction by desktop group.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SendTrigger

Queue command for delivery until this particular event occurs. Valid values are NextContact, Broker, LogOn, Logoff, Disconnect and Reconnect.

Type:	MachineCommandTrigger
Position:	Named
Default value:	Default value is 'NextContact'so the command is sent during the next communication with the desktop.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SendDeadline

Automatically cancel the command if it not delivered before the specified time span passes.

Type:	TimeSpan
Position:	Named
Default value:	Command expires after 24 hours.
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

No parameter is accepted from the input pipeline.

Outputs

Citrix.Broker.Admin.SDK.MachineCommand

New command that was added to the command queue.

Citrix.Broker.Admin.SDK.MachineSynchronousCommandResponse

When the Synchronous option is used, the command is immediately sent to the specified machine and processed. The SDK object returned describes the command and the result of this command processing.

Notes

Commands are subject to delegated administration restrictions based on the desktop group, category and command name.

Related Links

- Get-BrokerMachineCommand
- Remove-BrokerMachineCommand

New-BrokerMachineConfiguration

March 11, 2024

Creates a new machine configuration associated with an existing configuration slot.

Syntax

```
New-BrokerMachineConfiguration
   -ConfigurationSlotUid <Int32>
   [-Description <String>]
   -LeafName <String>
   -Policy <Byte[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a new machine configuration containing settings that match the SettingsGroup of the associated configuration slot. This machine configuration can then be applied to a desktop group to have the settings applied to machines in that group.

The SettingsGroup of the configuration slot restricts the permitted settings. Use the SDK snap-in that matches the SettingsGroup to create the encoded settings data.

Examples

EXAMPLE 1

Creates a new configuration named "%SlotName%\Finance Department"where %SlotName% is the name of the configuration slot having the Uid $csUid. The encoded settings in the $policy variable must match the SettingsGroup of the configuration slot having the Uid $csUid.

```
New-BrokerMachineConfiguration -LeafName "Finance Department" -
    Description "Finance Dept. User Profile Management policy" -Policy
    $policy -ConfigurationSlotUid $csUid
```

Parameters

-ConfigurationSlotUid

Unique identifier of the configuration slot to associate with this machine configuration.

Type:	Int32
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LeafName

Name of the new machine configuration. This must be unique amongst the machine configurations associated with the same configuration slot.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Policy

A binary array of encoded settings (policy) data created with the SDK snap-in that matches the SettingsGroup of the configuration slot.

Type:	Byte[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Description of the new machine configuration.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.MachineConfiguration

New-BrokerMachineConfiguration returns the newly created configuration

Notes

Delegated Administration can be used to restrict the configuration slots that an administrator can use and hence which components of the system that can be configured.

Related Links

- Get-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

New-BrokerPowerTimeScheme

March 11, 2024

Creates a new power time scheme for a desktop group.

Syntax

```
New-BrokerPowerTimeScheme
   -DaysOfWeek <TimeSchemeDays>
   -DesktopGroupUid <Int32>
   [-DisplayName <String>]
```



```
    [-Name] <String>
    [-PeakHalfHours <Boolean[]>]
    [-PeakHours <Boolean[]>]
    [-PoolSize <Int32[]>]
    [-PoolSizeHalfHours <Int32[]>]
    [-PoolUsingPercentage <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerPowerTimeScheme cmdlet adds a new power time scheme to be associated with a desktop group. The power time scheme must relate to days of the week that are not already covered by an existing power time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

See 'help about_Broker_PowerManagement'for a detailed description of the power policy mechanism and pool size management.

Examples

EXAMPLE 1

Creates a new scheme attached to the desktop group whose UID value is 3. This new scheme covers the weekend and Monday and Tuesday, and defines 'peak'hours as 8am to 6:30pm, with all other times being 'off-peak'. No pool size values are supplied, so all size values for all the hours default to -1.

```
New-BrokerPowerTimeScheme -Name 'First Half Week' -DaysOfWeek Weekend,
    Monday,Tuesday -DesktopGroupUid 3 -PeakHalfHours (0..47 | %{
  if ($_ -lt 16 -or $_ -gt 37) {
  $false  }
  else {
  $true  }
   }
  )
```


Parameters

-Name

Specifies the administrative name of the new power time scheme. Each scheme must have a name which is unique within the site.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DaysOfWeek

Specifies the pattern of days of the week that the power time scheme covers.

Valid values are (singly or a list of) Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday, Weekdays and Weekend.

Type:	TimeSchemeDays
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DesktopGroupUid

Specifies the desktop group that the power time scheme applies to.

Type:	Int32
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DisplayName

Specifies the name of the new power time scheme as displayed in the DesktopStudio console. Each scheme associated with a desktop group must have a display name which is unique within its desktop group, although the same display name can be used on power schemes for different desktop groups.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PeakHalfHours

A set of 48 boolean flag values, one for each half hour of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. If the flag is $true it means that the associated half hour of the day is considered a peak time; if $false it means that it is considered off-peak.

Type:	Boolean[]
Position:	Named
Default value:	48 $false values, meaning all half hours are off-peak
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PeakHours

A set of 24 boolean flag values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag is $true it means that the associated hour of the day is considered a peak time; if $false it means that it is considered off-peak.

Type:	Boolean[]
Position:	Named
Default value:	24 $false values, meaning all hours are off-peak
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PoolSize

For single-session desktop groups, a set of 24 integer values, one for each hour of the day. The first value in the array relates to midnight to 00:59, the next one to 1 AM to 01:59 and so on, with the last array element relating to 11 PM to 11:59. For multi-session desktop groups, a set of 48 integer values, one for each half hour of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such hours.

Type:	Int32[]
Position:	Named
Default value:	24 (or 48 for multi-session desktop groups) values of '-1', meaning no pool size management is to be performed
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PoolSizeHalfHours

A set of 48 integer values, one for each half hour of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. The value defines the number of machines (either as an absolute number or a percentage of the machines in the desktop group) that are to be maintained in a running state, whether they are in use or not. A value of -1 has special meaning: pool size management does not apply during such half hours.

Type:	Int32[]
Position:	Named
Default value:	48 values of '-1', meaning no pool size management is to be performed
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PoolUsingPercentage

A boolean flag to indicate whether the integer values in the pool size array are to be treated as absolute values (if this value is $false) or as percentages of the number of machines in the desktop group (if this value is $true).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.PowerTimeScheme

New-BrokerPowerTimeScheme returns the newly created power time scheme.

Related Links

- about_Broker_PowerManagement
- Get-BrokerPowerTimeScheme
- Set-BrokerPowerTimeScheme

- Remove-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme

New-BrokerRebootSchedule

March 11, 2024

Creates a new reboot schedule for a desktop group.

Syntax

```
New-BrokerRebootSchedule
   [-Day <RebootScheduleDays>]
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   -RebootDuration <Int32>
   [-StartTime <TimeSpan>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerRebootSchedule
   [-Day <RebootScheduleDays>]
   [-DesktopGroupName] <String>
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   -RebootDuration <Int32>
   [-StartTime <TimeSpan>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerRebootSchedule cmdlet is used to define a reboot schedule for a desktop group.

Examples

EXAMPLE 1

Schedules the machines in the desktop group named ‘BankTellers’to be rebooted every night between 2 AM and 4 AM.

```
New-BrokerRebootSchedule -DesktopGroupName BankTellers -Frequency Daily
    -StartTime "02:00" -Enabled $true -RebootDuration 120
```

EXAMPLE 2

Schedules the machines in the desktop group having Uid 17 to be rebooted every Saturday night between 1 AM and 5 AM. Ten minutes prior to rebooting, each machine will display a message box with the title “WARNING: Reboot pending”and message “Save your work”in every user session.

```
New-BrokerRebootSchedule -DesktopGroupUid 17 -Frequency Weekly -Day
    Saturday -StartTime "01:00" -Enabled $true -RebootDuration 240 -
    WarningTitle "WARNING: Reboot pending" -WarningMessage "Save your
    work" -WarningDuration 10
```

EXAMPLE 3

Schedules the machines in the desktop group named ‘BankTellers’to be rebooted every night between 3 AM and 5 AM. Fifteen, ten and five minutes prior to rebooting each machine, the message “Rebooting in %m% minutes.”will be displayed in each user session with the pattern ‘%m%’replaced with the number of minutes until the reboot.

```
New-BrokerRebootSchedule -DesktopGroupName BankTellers -Frequency Daily
    -StartTime "03:00" -Enabled $true -RebootDuration 120 -
    WarningMessage "Rebooting in %m% minutes." -WarningDuration 15 -
    WarningRepeatInterval 5
```

Parameters

-DesktopGroupName

The name of the desktop group that this reboot schedule is applied to.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DesktopGroupUid

The Uid of the desktop group that this reboot schedule is applied to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Day

For weekly schedules, the day of the week on which the scheduled reboot-cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the new reboot schedule is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Frequency

Frequency with which this schedule runs (either Weekly or Daily).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RebootSchedule

Related Links

- Get-BrokerRebootSchedule
- Set-BrokerRebootSchedule
- Remove-BrokerRebootSchedule
- Start-BrokerRebootCycle

New-BrokerRebootScheduleV2

March 11, 2024

Creates a new reboot schedule for a desktop group.

Syntax

```
New-BrokerRebootScheduleV2
   [-Day <RebootScheduleDays>]
   [-DayInMonth <RebootScheduleDays>]
   [-Description <String>]
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   [-FrequencyFactor <Int32>]
   [-IgnoreMaintenanceMode <Boolean>]
   [-MaxOvertimeStartMins <Int32>]
   [-Name] <String>
   -RebootDuration <Int32>
   [-RestrictToTag <String>]
   [-StartDate <String>]
   [-StartTime <TimeSpan>]
   [-UseNaturalReboot <Boolean>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-WeekInMonth <RebootScheduleWeeks>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerRebootScheduleV2
   [-Day <RebootScheduleDays>]
   [-DayInMonth <RebootScheduleDays>]
   [-Description <String>]
   -DesktopGroupName <String>
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   [-FrequencyFactor <Int32>]
   [-IgnoreMaintenanceMode <Boolean>]
   [-MaxOvertimeStartMins <Int32>]
   [-Name] <String>
   -RebootDuration <Int32>
   [-RestrictToTag <String>]
   [-StartDate <String>]
   [-StartTime <TimeSpan>]
   [-UseNaturalReboot <Boolean>]
```



```
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-WeekInMonth <RebootScheduleWeeks>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerRebootScheduleV2 cmdlet is used to define a reboot schedule for a desktop group.

Examples

EXAMPLE 1

Schedules the machines in the desktop group named ‘BankTellers’to be rebooted every night between 2 AM and 4 AM.

```
New-BrokerRebootScheduleV2 -Name BankTellers-DailyReboot -
    DesktopGroupName BankTellers -Frequency Daily -StartTime "02:00" -
    Enabled $true -RebootDuration 120
```

EXAMPLE 2

Schedules the machines in the desktop group having Uid 17 to be rebooted every Saturday early morning between 1 AM and 5 AM. Ten minutes prior to rebooting, each machine will display a message box with the title “WARNING: Reboot pending”and message “Save your work”in every user session.

```
New-BrokerRebootScheduleV2 -Name 'Saturday reboots' -DesktopGroupUid 17
     -Frequency Weekly -Day Saturday -StartTime "01:00" -Enabled $true -
    RebootDuration 240 -WarningTitle "WARNING: Reboot pending" -
    WarningMessage "Save your work" -WarningDuration 10
```

EXAMPLE 3

Schedules the machines in the desktop group having Uid 17 to be rebooted Saturday and Sunday early morning between 1 AM and 5 AM for every four weeks starting from Feb 6, 2021. Ten minutes prior to rebooting, each machine will display a message box with the title “WARNING: Reboot pending”and message “Save your work”in every user session.


```
New-BrokerRebootScheduleV2 -Name 'Saturday reboots' -DesktopGroupUid 17
    -Frequency Weekly -FrequencyFactor 4 -Day "Saturday,Sunday" -
    StartDate "2021-02-03" -StartTime "01:00" -Enabled $true -
    RebootDuration 240 -WarningTitle "WARNING: Reboot pending" -
    WarningMessage "Save your work" -WarningDuration 10
```

EXAMPLE 4

Schedules the machines in the desktop group having Uid 17 to be rebooted the last Saturday of the month early morning between 1 AM and 5 AM for every month. Ten minutes prior to rebooting, each machine will display a message box with the title "WARNING: Reboot pending"and message "Save your work"in every user session.

```
New-BrokerRebootScheduleV2 -Name 'Monthly reboots' -DesktopGroupUid 17
    -Frequency Monthly -DayInMonth Saturday -WeekInMonth Last -StartTime
    "01:00" -Enabled $true -RebootDuration 240 -WarningTitle "WARNING:
    Reboot pending" -WarningMessage "Save your work" -WarningDuration 10
```

EXAMPLE 5

Schedules the machines in the desktop group having Uid 17 to be rebooted the first Sunday of the month early morning between 1 AM and 5 AM for every two months, starting from Feb 7, 2021 which is the first Sunday of Feburary, 2021. Ten minutes prior to rebooting, each machine will display a message box with the title "WARNING: Reboot pending"and message "Save your work"in every user session.

```
New-BrokerRebootScheduleV2 -Name 'Monthly reboots' -DesktopGroupUid 17
    -Frequency Monthly -FrequencyFactor 2 -StartDate "2021-02-03" -
    DayInMonth Sunday -WeekInMonth First -StartTime "01:00" -Enabled
    $true -RebootDuration 240 -WarningTitle "WARNING: Reboot pending" -
    WarningMessage "Save your work" -WarningDuration 10
```

EXAMPLE 6

Schedules the machines in the desktop group named 'BankTellers'to be rebooted every night between 3 AM and 5 AM. Fifteen, ten and five minutes prior to rebooting each machine, the message "Rebooting in %m% minutes."will be displayed in each user session with the pattern '%m%'replaced with the number of minutes until the reboot.

```
New-BrokerRebootScheduleV2 -Name BankTellers-DailyReboot -
    DesktopGroupName BankTellers -Frequency Daily -StartTime "03:00" -
    Enabled $true -RebootDuration 120 -WarningMessage "Rebooting in %m%
    minutes." -WarningDuration 15 -WarningRepeatInterval 5
```


EXAMPLE 7

Schedules only machines with the tag ‘Daily Reboot’in the desktop group named ‘BankTellers’to be rebooted every night between 3 AM and 5 AM. Fifteen, ten and five minutes prior to rebooting each machine, the message “Rebooting in %m% minutes.”will be displayed in each user session with the pattern ‘%m%’replaced with the number of minutes until the reboot.

```
New-BrokerRebootScheduleV2 -Name BankTellers-DailyReboot -
    DesktopGroupName BankTellers -Frequency Daily -StartTime "03:00" -
    Enabled $true -RebootDuration 120 -WarningMessage "Rebooting in %m%
    minutes." -WarningDuration 15 -WarningRepeatInterval 5 -
    RestrictToTag 'Daily Reboot'
```

Parameters

-Name

The name of the new reboot schedule.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group that this reboot schedule is applied to.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

The Uid of the desktop group that this reboot schedule is applied to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Day

For weekly schedules, the days of the week on which the scheduled reboot-cycle starts (one or more of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-DayInMonth

For monthly schedules, the day in the month on which the scheduled reboot-cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

An optional description for the reboot schedule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Boolean that indicates if the new reboot schedule is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Frequency

Frequency with which this schedule runs (either Weekly or Daily or Monthly).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-FrequencyFactor

The frequency factor for the reboot schedule. It indicates how often the schedule should run with respect to the frequency. For example, if the FrequencyFactor is set to 2, it means every two days from the StartDate when the Frequency is Daily, every two weeks from StartDate when the Frequency is Weekly, and similarly for monthly. It defaults to 1 if not provided. StartDate property needs to be provided if the frequency factor is greater than 1.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IgnoreMaintenanceMode

Boolean value to reboot machines in maintenance mode

Type:	Boolean
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-RestrictToTag

If set the reboot schedule only applies to machines in the desktop group with the specified tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-StartDate

The date on which the first schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD). The actual start date to match the frequency pattern need not be provided. For example, if today is Monday and the schedule is set to run every Sunday for every four weeks, there is no need to identify the date on which Sunday falls after four weeks. Today's date ((Get-Date).Date.ToString('yyyy-MM-dd')) can be provided and the service would adjust the start date to reflect the actual date i.e, the Sunday four weeks from today. Get-BrokerRebootScheduleV2 cmdlet will reflect the actual start date.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UseNaturalReboot

Boolean value indicating whether the machines should reboot whenever they happen to have no sessions, rather than at equally spaced times within the cycle duration.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-WeekInMonth

For monthly schedules, the week in the month on which the scheduled reboot-cycle starts (one of First, Second, Third, Fourth, Last).

Type:	RebootScheduleWeeks
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RebootScheduleV2

Related Links

- Get-BrokerRebootScheduleV2

- Set-BrokerRebootScheduleV2
- Remove-BrokerRebootScheduleV2
- Rename-BrokerRebootScheduleV2
- Start-BrokerRebootCycle

New-BrokerRemotePCAccount

March 11, 2024

Create a new RemotePCAccount.

Syntax

```
New-BrokerRemotePCAccount
   [-AllowSubfolderMatches <Boolean>]
   -CatalogUid <Int32>
   [-MachinesExcluded <String[]>]
   [-MachinesIncluded <String[]>]
   -OU <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Create a new RemotePCAccount. A RemotePCAccount defines machine filters to support Remote PC automation adding unconfigured machines to catalogs.

Examples

EXAMPLE 1

Create a RemotePCAccount that adds unconfigured machines with computer objects in MyOU, into catalog 42.

```
New-BrokerRemotePCAccount -OU 'ou=MyOU,dc=MyDomain,dc=com' -CatalogUid
    42
```


EXAMPLE 2

Create a RemotePCAccount matching unconfigured machines from DOMAIN1, except those with host-names containing JOHNDOE, and add them to catalog 42.

```
New-BrokerRemotePCAccount -OU 'any' -CatalogUid 42 -MachinesIncluded @(
    'DOMAIN1\*') -MachinesExcluded @('DOMAIN1\*JOHNDOE*')
```

EXAMPLE 3

Create a RemotePCAccount that matches any unconfigured machine, causing automation to add matching machines to catalog 42.

```
New-BrokerRemotePCAccount -OU 'any' -CatalogUid 42
```

Parameters

-CatalogUid

Specifies the catalog which Remote PC automation adds an unconfigured machine to if it matches this RemotePCAccount.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-OU

Specifies the DN of an AD container, or has the special value ‘any’.

When an AD container is specified a machine may only match with the RemotePCAccount when the AD computer object is located relative to the OU.

When ‘any’is specified the location of the AD computer object is ignored for purposes of matching this RemotePCAccount. The machine must still meet the MachinesIncluded and MachinesExcluded filters for a match to occur.

In the event that a machine matches with multiple RemotePCAccounts then the RemotePCAccount OU with the longest canonical name takes precedence. The special 'any'OU is treated as lowest priority.

Note that the OU value of every RemotePCAccount must be unique, and this includes only one 'any' entry being permitted.

Type:	String
Position:	Named
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AllowSubfolderMatches

When true a machine matches this RemotePCAccount if the AD computer object exists within the container specified by the OU property, or within a child container of the OU.

When false the AD computer object only matches if it exists directly in the AD container specified by the OU property.

This property is not meaningful when OU has the special value 'any'.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachinesExcluded

MachinesExcluded specifies a set of strings that can include asterisk wildcards. If a machine name matches any entries in MachinesExcluded then it cannot match with this RemotePCAccount regardless of whether there is a MachinesIncluded match.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

DOMAIN1\M*

DOMAIN*\M*

\M

Type:	String[]
Position:	Named
Default value:	@()
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachinesIncluded

MachinesIncluded specifies a set of strings that can include asterisk wildcards. A machine may only match with this RemotePCAccount if it matches a MachinesIncluded entry and does not match any MachinesExcluded entries.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

DOMAIN1\M*

DOMAIN*\M*

\M

Type:	String[]
Position:	Named
Default value:	@('*')
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.RemotePCAccount

The newly created RemotePCAccount.

Related Links

- about_Broker_RemotePC
- Get-BrokerRemotePCAccount
- Set-BrokerRemotePCAccount
- Remove-BrokerRemotePCAccount

New-BrokerSessionLinger

March 11, 2024

Creates a new session linger setting for a desktop group.

Syntax

```
New-BrokerSessionLinger
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerSessionLinger
   [-DesktopGroupName] <String>
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerSessionLinger cmdlet is used to define a session linger setting for a desktop group.

Note that each desktop group can only have a single session linger setting. Session lingering only applies to application sessions.

Examples

EXAMPLE 1

Creates a new session linger setting with a disconnect timer of 30 minutes and terminate timer of 1 hour.

```
New-BrokerSessionLinger -DesktopGroupName test -Enabled $true -
    MaxTimeBeforeDisconnect 0:30 -MaxTimeBeforeTerminate 1:00
```

Parameters

-DesktopGroupName

The name of the desktop group that this linger setting is applied to.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

The Uid of the desktop group that this linger setting is applied to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the new session linger is enabled.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Specifies the average load threshold across the desktop group. When the threshold hits, lingering sessions across the group be terminated to reduce load. Sessions that have been lingering the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Specifies the maximum load threshold per machine in the desktop group. When the threshold hits, lingering sessions on each loaded machine will be terminated to reduce load. Sessions that have been lingering the longest will be chosen first.

Type:	Int32
Position:	Named

Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Specifies the time by which a lingering session will be disconnected. The disconnect timer is optional, but when specified the terminate timer needs to be also set. The disconnect time cannot be greater than the terminate time. When the disconnect and terminate times are the same, the terminate timer takes precedence. The disconnect timer needs to be paired with a session termination condition like the terminate timer or one of load threshold settings.

Type:	TimeSpan
Position:	Named
Default value:	15 minutes
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Specifies the time by which a lingering session will be terminated. The terminate timer is not optional when timers are configured. When the disconnect and terminate times are the same, the terminate timer takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	8 hours
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UserFilterEnabled

Specifies whether the session linger's user filter is enabled or disabled. Where the user filter is enabled, lingering is enabled only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

Citrix.Broker.Admin.SDK.SessionLinger

New-BrokerSessionLinger returns a session linger object.

Related Links

- Get-BrokerSessionLinger
- Set-BrokerSessionLinger
- Remove-BrokerSessionLinger

New-BrokerSessionPreLaunch

March 11, 2024

Creates a new session pre-launch setting for a desktop group.

Syntax

```
New-BrokerSessionPreLaunch
   -DesktopGroupUid <Int32>
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
```



```
    [-UserFilterEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-BrokerSessionPreLaunch
    [-DesktopGroupName] <String>
    [-Enabled <Boolean>]
    [-MaxAverageLoadThreshold <Int32>]
    [-MaxLoadPerMachineThreshold <Int32>]
    [-MaxTimeBeforeDisconnect <TimeSpan>]
    [-MaxTimeBeforeTerminate <TimeSpan>]
    [-UserFilterEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The New-BrokerSessionPreLaunch cmdlet is used to define a session pre-launch setting for a desktop group.

Note that each desktop group can only have a single session pre-launch setting. Session pre-launch only applies to application sessions.

Examples

EXAMPLE 1

Creates a new session pre-launch setting with a disconnect timer of 30 minutes and terminate timer of 1 hour.

```
New-BrokerSessionPreLaunch -DesktopGroupName test -Enabled $true -
    MaxTimeBeforeDisconnect 0:30 -MaxTimeBeforeTerminate 1:00
```

Parameters

-DesktopGroupName

The name of the desktop group that this pre-launch setting is applied to.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DesktopGroupUid

The Uid of the desktop group that this pre-launch setting is applied to.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the new session pre-launch is enabled.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Specifies the average load threshold across the desktop group. When the threshold hits, pre-launched sessions across the group be terminated to reduce load. Sessions that have been pre-launched the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Specifies the maximum load threshold per machine in the desktop group. When the threshold hits, pre-launched sessions on each loaded machine will be terminated to reduce load. Sessions that have been pre-launched the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Specifies the time by which a pre-launched session will be disconnected. The disconnect timer is optional, but when specified the terminate timer needs to be also set. The disconnect time cannot be greater than the terminate time. When the disconnect and terminate times are the same, the terminate timer takes precedence. The disconnect timer needs to be paired with a session termination condition like the terminate timer or one of load threshold settings.

Type:	TimeSpan
Position:	Named
Default value:	15 minutes
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Specifies the time by which a pre-launched session will be terminated. The terminate timer is not optional when timers are configured. When the disconnect and terminate times are the same, the terminate timer takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	8 hours
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UserFilterEnabled

Specifies whether the session pre-launch's user filter is enabled or disabled. Where the user filter is enabled, pre-launch is enabled only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

Citrix.Broker.Admin.SDK.SessionPreLaunch

New-BrokerSessionPreLaunch returns a session pre-launch object.

Related Links

- Get-BrokerSessionPreLaunch

- Set-BrokerSessionPreLaunch
- Remove-BrokerSessionPreLaunch

New-BrokerStorefrontAddress

March 11, 2024

Creates a new StoreFront address configuration, specifying a single address.

Syntax

```
New-BrokerStorefrontAddress
   -Name <String>
   -Url <String>
   -Enabled <Boolean>
   -Description <String>
   [<CommonParameters>]
```

Description

Use this command when you want to create a new StoreFront configuration byte array from scratch, rather than modifying an existing one. You must define the URL for the StoreFront, and some additional details.

This command does not, by itself, have any persistent effects within XenDesktop. To make the change persistent, the new configuration byte array must first be transformed into a machine configuration within the Citrix Broker Service. To do this, use the New-BrokerMachineConfiguration command. You can then use the Add-BrokerMachineConfiguration and Set-BrokerMachineConfiguration commands to fully associate the new configuration with a delivery group.

Examples

EXAMPLE 1

This example shows a new configuration byte array being created to specify a single StoreFront address. The configuration byte array is then provided as input to the Get-BrokerStorefrontAddress command, which interprets and outputs the same fields.


```
$configuration = New-BrokerStorefrontAddress -Url "https://mysite.com/
    Citrix/StoreWeb" -Description "This StoreFront delivers my corporate
     applications" -Name "StoreFront1" -Enabled $true

Get-BrokerStorefrontAddress -ByteArray $configuration

Name               Url                                        Enabled
    Description
----               ---                                        -------
    -----------
StoreFront1        https://mysite.com/Citrix/StoreWeb           True
    This StoreFront delivers my corporate applications.
```

Parameters

-Name

Specifies the name of the new StoreFront.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Url

Specifies the URL to the StoreFront, such as “https://mysite.com/Citrix/StoreWeb”.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Specifies if the new StoreFront address should be enabled for user access.

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Specifies a human-readable description of the new StoreFront.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

The new configuration set, with all of the given modifications applied.

Related Links

- Add-BrokerStorefrontAddress
- Get-BrokerStorefrontAddress
- New-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration

New-BrokerTag

March 11, 2024

Creates a new tag.

Syntax

```
New-BrokerTag
   [-Description <String>]
   [-Name] <String>
   [-Scope <String[]>]
   [-UUID <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a tag that can be associated with other objects using Add-BrokerTag.

Examples

EXAMPLE 1

Creates a new tag with name 'Tag1'.


```
New-BrokerTag -Name 'Tag1'
```

EXAMPLE 2

Creates a new tag with name ‘Tag2’and associates it with machine DOMAIN\Machine.

```
New-BrokerTag 'Tag2' | Add-BrokerTag -Machine DOMAIN\Machine
```

Parameters

-Name

Specifies a name for the new tag.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

A description for the tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Scope

Specifies the name of the delegated administration scope to which the tag should belong.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-UUID

Specifies a UUID for the new tag. When not specified, a UUID is automatically assigned.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.Tag

Outputs the generated tag.

Related Links

- Add-BrokerTag
- Get-BrokerTag
- Remove-BrokerTag
- Rename-BrokerTag
- Set-BrokerTag

New-BrokerUniversalClaim

March 11, 2024

A claim can be an arbitrary string up to 450 characters in length. Each Claim is mapped to a Virtual SID. Virtual SIDs are generated by either taking a SHA256 hash of the claim, and storing the hash bytes in a SID format, or in the case of a SID string being used as a Claim, the SID is copied to the VirtualSid.

Syntax

```
New-BrokerUniversalClaim
   -Claim <String>
   [-DirectoryContext] <String>
   [-UniversalClaimsTenantContext <String>]
   [-VirtualSid] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a new UniversalClaim object.

Examples

EXAMPLE 1

Creates a new UniversalClaim mapping

```
New-BrokerUniversalClaim -Claim AD:... -VirtualSid S-1-...1234 -
   DirectoryContext {
 .. }
```

Parameters

-VirtualSid

The VirtualSid for this UniversalClaim mapping.

Type: String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Claim

The Claim for this UniversalClaim mapping.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DirectoryContext

The DirectoryContext associated with this UniversalClaim mapping.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-UniversalClaimsTenantContext

The UnivesalClaimsTenantContext for this UniversalClaim mapping.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.UniversalClaim

Get-BrokerUniversalClaim returns an object for each matching broker UniversalClaim mapping.

Related Links

New-BrokerUser

March 11, 2024

Creates a new broker user object

Syntax

```
New-BrokerUser
   [-SID] <SecurityIdentifier>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerUser
   [-Name] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerUser cmdlet creates a new broker object to represent a user identity (or the identity of a group of users). The object is created local to the PowerShell environment in which the cmdlet is run; no new user object is created in the broker configuration, unless the object is added to another broker object, such as a machine or a desktop. For details, see Add-BrokerUser.

The identity of the user or group must be specified using either the Name or SID parameter

Examples

EXAMPLE 1

Create a broker user object for the specified user.

```
$user = New-BrokerUser DOMAIN\UserName
```

EXAMPLE 2

Create a broker user object for the specified user.

```
$user = New-BrokerUser -SID S
    -1-5-23-1763203430-193137401-908696819-3450
```

Parameters

-SID

The SID of the user or group

Type:	SecurityIdentifier
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

The name of the user or group

Type:	String
Position:	2
Default value:	Null
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.User

The broker user object

Notes

Typically, broker user objects are created implicitly using the

Add-BrokerUser cmdlet with a user name or SID.

Related Links

- Add-BrokerUser
- Get-BrokerUser
- Remove-BrokerUser

New-BrokerUserZonePreference

March 11, 2024

Creates a zone preference for a user/group account in this site

Syntax

```
New-BrokerUserZonePreference
   -HomeZoneUid <Guid>
   [-Name] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-BrokerUserZonePreference
   -HomeZoneUid <Guid>
   [-SID] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The New-BrokerUserZonePreference cmdlet specifies a preferred home zone for resources launched using the specified user/group account.

Subject to the configuration of the desktop groups in use, and the availability of machines in the preferred zone, desktops and applications are launched using machines in that zone where possible.

Examples

EXAMPLE 1

Sets the preferred zone for resources launched by members of the EMEA\sales group account.

```
$zp = New-BrokerUserZonePreference EMEA\sales -HomeZoneUid 2E885C02-6
   B65-47AA-8B03-E855BE2FF7D7
```


Parameters

-Name

Name of the user/group account with which the new home zone preference is to be associated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-SID

SID of the user/group account with which the new home zone preference is to be associated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-HomeZoneUid

The home zone preference to be associated with the user/group account.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.UserZonePreference

New-BrokerUserZonePreference returns the newly created zone preference object.

Related Links

- Get-BrokerUserZonePreference
- Set-BrokerUserZonePreference
- Remove-BrokerUserZonePreference
- Get-BrokerUser

New-BrokerXmlServiceKey

March 11, 2024

Generate a new 256 bit, base64 encoded, XML Service Key

Syntax

```
New-BrokerXmlServiceKey
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

To be used with the Set-BrokerSite Cmdlet to set the two XmlServiceKey properties.

Examples

EXAMPLE 1

Sets XmlServiceKey1 in the Site settings

```
$xmlServiceKey1 = New-BrokerXmlServiceKey
Set-BrokerSite -XmlServiceKey1 $xmlServiceKey1
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

Related Links

Remove-BrokerAccessPolicyRule

March 11, 2024

Deletes a rule from the site's access policy.

Syntax

```
Remove-BrokerAccessPolicyRule
      [-InputObject] <AccessPolicyRule[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-BrokerAccessPolicyRule
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAccessPolicyRule cmdlet deletes a rule from the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Deleting a rule does not affect existing user sessions, but it may result in users being unable to launch new sessions, or reconnect to disconnected sessions if access to the desktop group delivering those sessions was granted by the deleted rule.

Examples

EXAMPLE 1

Deletes the access policy rule called Temp Staff. Existing sessions are not affected, but if access was granted by the deleted rule users may be unable to reconnect to sessions if they are subsequently disconnected.

```
Remove-BrokerAccessPolicyRule 'Temp Staff'
```

EXAMPLE 2

Deletes all access policy rules explicitly granting user SALES\johndoe access to any desktop group in the site. Any existing desktop sessions for the user are not affected. The user may still be able to access site resources by access policy rules that grant access through group membership or non-user-based connection filters.

```
Get-BrokerAccessPolicyRule -IncludedUsers sales\johndoe | Remove-
    BrokerAccessPolicyRule
```


Parameters

-InputObject

The access policy rule to be deleted.

Type:	AccessPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the access policy rule to be deleted.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AccessPolicyRule

The access policy rule to be deleted.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- New-BrokerAccessPolicyRule
- Get-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule

Remove-BrokerAccessPolicyRuleMetadata

March 11, 2024

Deletes AccessPolicyRule Metadata from the AccessPolicyRule objects

Syntax

```
Remove-BrokerAccessPolicyRuleMetadata
      [-InputObject] <AccessPolicyRule[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAccessPolicyRuleMetadata cmdlet deletes Metadata from the AccessPolicyRule objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the AccessPolicyRule whose instance is pointed by $obj-Uid

```
Remove-BrokerAccessPolicyRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the AccessPolicyRule in the site

```
Get-BrokerAccessPolicyRule | Remove-BrokerAccessPolicyRuleMetadata -
    Name "MyMetadataName"
```


Parameters

-InputObject

Specifies the AccessPolicyRule object's instance whose Metadata is to be deleted.

Type:	AccessPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAccessPolicyRule

You can pipe the AccessPolicyRule to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerAdminFolder

March 11, 2024

Removes an admin folder.

Syntax

```
Remove-BrokerAdminFolder
      [-InputObject] <AdminFolder[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerAdminFolder
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAdminFolder cmdlet removes an existing admin folder.

Remove-BrokerAdminFolder will not remove a folder if it contains any other objects (e.g. sub-folders or applications).

Examples

EXAMPLE 1

Removes the folder called F2 within the folder F1\

```
Remove-BrokerAdminFolder F1\F2\
```

Parameters

-InputObject

Identifies the folder to remove

Type:	AdminFolder[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name pattern of folder(s) to remove

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AdminFolder

The admin folder objects can be specified as input.

Outputs

None

This cmdlet does not return any output.

Related Links

- Get-BrokerAdminFolder
- New-BrokerAdminFolder

Remove-BrokerAdminFolderMetadata

March 11, 2024

Deletes AdminFolder Metadata from the AdminFolder objects

Syntax

```
Remove-BrokerAdminFolderMetadata
       [-InputObject] <AdminFolder[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Remove-BrokerAdminFolderMetadata cmdlet deletes Metadata from the AdminFolder objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the AdminFolder whose instance is pointed by $obj-Uid

```
Remove-BrokerAdminFolderMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the AdminFolder in the site

```
Get-BrokerAdminFolder | Remove-BrokerAdminFolderMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the AdminFolder object’s instance whose Metadata is to be deleted.

Type:	AdminFolder[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAdminFolder

You can pipe the AdminFolder to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerAppAssignmentPolicyRule

March 11, 2024

Deletes an application rule from the site's assignment policy.

Syntax

```
Remove-BrokerAppAssignmentPolicyRule
      [-InputObject] <AppAssignmentPolicyRule[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerAppAssignmentPolicyRule
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAppAssignmentPolicyRule cmdlet deletes an application rule from the site's assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

Deleting an application rule does not remove machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Examples

EXAMPLE 1

Deletes the application rule called Temp Staff from the assignment policy. Access to machines already assigned by this rule is not affected in any way.

```
1 Remove-BrokerAppAssignmentPolicyRule 'Temp Staff'
```

EXAMPLE 2

Deletes the application rule for the Sales Support desktop group from the site's assignment policy. This prevents any further machine assignments being made from this group, but it does not affect existing assignments made by the rule.

```
1 $dg = Get-BrokerDesktopGroup 'Sales Support'
2 Get-BrokerAppAssignmentPolicyRule -DesktopGroupUid $dg.Uid | Remove-
     BrokerAppAssignmentPolicyRule
```

Parameters

-InputObject

The application rule to be deleted from the assignment policy.

Type:	AppAssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the application rule to be deleted from the assignment policy.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

The application rule to be deleted from the assignment policy.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppAssignmentPolicyRule
- Get-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule

Remove-BrokerAppEntitlementPolicyRule

March 11, 2024

Deletes an application rule from the site's entitlement policy.

Syntax

```
Remove-BrokerAppEntitlementPolicyRule
       [-InputObject] <AppEntitlementPolicyRule[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerAppEntitlementPolicyRule
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Remove-BrokerAppEntitlementPolicyRule cmdlet deletes an application rule from the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

Deleting a rule does not affect existing sessions launched using the rule, but users cannot reconnect to those sessions if they are subsequently disconnected.

Examples

EXAMPLE 1

Deletes the application rule called Temp Workers from the entitlement policy rule. Existing application sessions launched using that rule are not affected, but users cannot reconnect to those sessions if they are subsequently disconnected.

```
1 Remove-BrokerAppEntitlementPolicyRule 'Temp Workers'
```

EXAMPLE 2

Deletes the application rule from the entitlement policy rule applied to the Customer Support desktop group. This effectively removes all access to the applications published from this group. Existing application sessions are not affected, but users cannot reconnect to those sessions if they are subsequently disconnected.

```
1 $dg = Get-BrokerDesktopGroup 'Customer Support'
2 Get-BrokerAppEntitlementPolicyRule -DesktopGroupUid $dg.Uid | Remove-
     BrokerAppEntitlementPolicyRule
```

Parameters

-InputObject

The application rule to be deleted from the entitlement policy.

Type:	AppEntitlementPolicyRule[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the application rule to be deleted from the entitlement policy.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

The application rule to be deleted from the entitlement policy.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppEntitlementPolicyRule
- Get-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule

Remove-BrokerApplication

March 11, 2024

Deletes one or more applications, or an association of an application.

Syntax


```
Remove-BrokerApplication
      [-InputObject] <Application[]>
      [-Force]
      [-ApplicationGroup <ApplicationGroup>]
      [-DesktopGroup <DesktopGroup>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerApplication
      [-Force]
      [-Name] <String>
      [-ApplicationGroup <ApplicationGroup>]
      [-DesktopGroup <DesktopGroup>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerApplication cmdlet deletes one or more applications, or you can use it to delete just the association of an application to a desktop group or application group.

Its usage dictates the behavior of the cmdlet. If only the application is specified as a parameter, then the cmdlet deletes the application. It also deletes any associations this application has with other objects, such as with access policy rules or desktop groups. More specifically, when an application is deleted the following happens:

- The association to any desktop groups is removed.
- The association to any application groups is removed.
- The association to any tags is removed.
- Any configured file-type association objects for this application are deleted.
- The association to any user accounts is removed.
- The association to any access session conditions is removed.
- The access policy rule object for this application, if one existed, is deleted.
- Finally, the application object itself is deleted.

Note that if the application is in use by a user then the application cannot be deleted.

If more than just the application is supplied as a parameter to the cmdlet (for instance, if a Desktop-Group object is also specified) then the application is not deleted. Instead, only the association from the application to that desktop group is removed.

Examples

EXAMPLE 1

This command deletes the application that has a BrowserName of "Notepad".

```
Remove-BrokerApplication "Notepad"
```

EXAMPLE 2

This command removes the association of the desktop group that has a name of "Private Desktop-Group"from the application that has a BrowserName of "Notepad". It does not otherwise modify the application.

```
$app = Get-BrokerApplication -BrowserName "Notepad"
$group = Get-BrokerDesktopGroup -Name "Private DesktopGroup"
Remove-BrokerApplication -InputObject $app -DesktopGroup $group
```

Parameters

-InputObject

Specifies the applications to delete. The Uid can also be substituted for the application objects.

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application to remove.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Force

Remove application even if it's in use. Removing an application that is currently in use, can potentially leave an application session containing no applications. If all the applications that are currently active in a disconnected application session are removed, the user will be unable to reconnect to the session. Forcing removal of an in-use application does not impact the actual session itself.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationGroup

Specifies the application group that this application should no longer be associated with. The Uid or Name can also be substituted for the application group object.

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Specifies the desktop group that this application should no longer be associated with. The Uid or Name can also be substituted for the desktop group object.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Application

The application objects can be specified as input.

Outputs

None

This cmdlet does not return any output.

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Add-BrokerApplication
- Get-BrokerApplication
- Rename-BrokerApplication
- Move-BrokerApplication
- Set-BrokerApplication

Remove-BrokerApplicationGroup

March 11, 2024

Remove application groups from the system, or break the association between an application group and a desktop group.

Syntax

```
Remove-BrokerApplicationGroup
       [-InputObject] <ApplicationGroup[]>
       [-Force]
       [-DesktopGroup <DesktopGroup>]
       [-LoggingId <Guid>]
```



```
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-BrokerApplicationGroup
        [-Force]
        [-Name] <String>
        [-DesktopGroup <DesktopGroup>]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This cmdlet has 2 functions:

- Break associations between application groups and desktop groups.
- Remove application groups from the system.

Associating an application group with a desktop group allows the applications that are members of that application group to be launched on machines that are members of the associated desktop group. Breaking that association means that those applications may no longer be launched on those machines.

To remove an application group from the system, you must first remove all of its applications. Use the Remove-BrokerApplication cmdlet to remove an application (either to remove it from an application group, or to remove it from the system entirely).

Examples

EXAMPLE 1

Remove the association between the 'Office'application group and the 'Windows10VDAs'desktop group. The 'Office'applications will no longer be launchable on the machines in the 'Windows10VDAs' desktop group.

```
Remove-BrokerApplicationGroup Office -DesktopGroup Windows10VDAs
```

EXAMPLE 2

Remove the 'Office'application group from the system altogether. You must remove all applications from the application group first.

```
Remove-BrokerApplicationGroup Office
```


Parameters

-InputObject

Specifies the application groups to remove.

Type:	ApplicationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Removes application groups whose name matches the given pattern.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Force

When this flag is specified, an application group may be removed from the system even if its Session-SharingEnabled property is false and one of its applications is still running in some session.

Such an application group cannot by default be removed, because doing so would allow for other applications to be launched into existing sessions. Either wait until the sessions associated with the application group have exited, or use the -Force flag.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroup

When this parameter is specified, the application groups are removed from the given desktop group. The desktop group may be specified either by its Uid or by its name.

When this parmeter is not specified, the application groups are removed from the system entirely.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ApplicationGroup

You can pipe application groups to Remove-BrokerApplicationGroup.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Applications
- Add-BrokerApplicationGroup
- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Rename-BrokerApplicationGroup
- Move-BrokerApplicationGroup
- Set-BrokerApplicationGroup
- Remove-BrokerApplication

Remove-BrokerApplicationGroupMetadata

March 11, 2024

Deletes ApplicationGroup Metadata from the ApplicationGroup objects

Syntax

```
Remove-BrokerApplicationGroupMetadata
      [-InputObject] <ApplicationGroup[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerApplicationGroupMetadata cmdlet deletes Metadata from the ApplicationGroup objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the ApplicationGroup whose instance is pointed by $obj-Uid

```
Remove-BrokerApplicationGroupMetadata -InputObject $obj-Uid -Name "
   MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the ApplicationGroup in the site

```
Get-BrokerApplicationGroup | Remove-BrokerApplicationGroupMetadata -
   Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the ApplicationGroup object's instance whose Metadata is to be deleted.

Type:	ApplicationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplicationGroup

You can pipe the ApplicationGroup to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerApplicationInstanceMetadata

March 11, 2024

Deletes ApplicationInstance Metadata from the ApplicationInstance objects

Syntax

```
Remove-BrokerApplicationInstanceMetadata
      [-InputObject] <ApplicationInstance[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

The Remove-BrokerApplicationInstanceMetadata cmdlet deletes Metadata from the ApplicationInstance objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the ApplicationInstance whose instance is pointed by $obj-Uid

```
Remove-BrokerApplicationInstanceMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the ApplicationInstance in the site

```
Get-BrokerApplicationInstance | Remove-
    BrokerApplicationInstanceMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the ApplicationInstance object's instance whose Metadata is to be deleted.

Type:	ApplicationInstance[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplicationInstance

You can pipe the ApplicationInstance to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerApplicationMetadata

March 11, 2024

Deletes Application Metadata from the Application objects

Syntax

```
Remove-BrokerApplicationMetadata
       [-InputObject] <Application[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerApplicationMetadata cmdlet deletes Metadata from the Application objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Application whose instance is pointed by $obj-Uid


```
Remove-BrokerApplicationMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Application in the site

```
Get-BrokerApplication | Remove-BrokerApplicationMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the Application object’s instance whose Metadata is to be deleted.

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplication

You can pipe the Application to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerAssignmentPolicyRule

March 11, 2024

Deletes a desktop rule from the site's assignment policy.

Syntax

```
Remove-BrokerAssignmentPolicyRule
      [-InputObject] <AssignmentPolicyRule[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerAssignmentPolicyRule
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAssignmentPolicyRule cmdlet deletes a desktop rule from the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Deleting a desktop rule does not remove machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Examples

EXAMPLE 1

Deletes the desktop rule called Temp Staff from the assignment policy. Access to machines already assigned by this rule is not affected in any way.

```
Remove-BrokerAssignmentPolicyRule 'Temp Staff'
```


EXAMPLE 2

Deletes all desktop rules for the Sales Support desktop group from the site's assignment policy. This prevents any further machine assignments being made from this group, but it does not affect existing assignments made by these rules.

```
$dg = Get-BrokerDesktopGroup 'Sales Support'
Get-BrokerAssignmentPolicyRule -DesktopGroupUid $dg.Uid | Remove-
    BrokerAssignmentPolicyRule
```

Parameters

-InputObject

The desktop rule to be deleted from the assignment policy.

Type:	AssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the desktop rule to be deleted from the assignment policy.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

The desktop rule to be deleted from the assignment policy.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_ErrorHandling
- New-BrokerAssignmentPolicyRule
- Get-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule

Remove-BrokerAssignmentPolicyRuleMetadata

March 11, 2024

Deletes AssignmentPolicyRule Metadata from the AssignmentPolicyRule objects

Syntax

```
Remove-BrokerAssignmentPolicyRuleMetadata
      [-InputObject] <AssignmentPolicyRule[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerAssignmentPolicyRuleMetadata cmdlet deletes Metadata from the Assignment-PolicyRule objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the AssignmentPolicyRule whose instance is pointed by $obj-Uid

```
Remove-BrokerAssignmentPolicyRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```


EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the AssignmentPolicyRule in the site

```
Get-BrokerAssignmentPolicyRule | Remove-
    BrokerAssignmentPolicyRuleMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the AssignmentPolicyRule object’s instance whose Metadata is to be deleted.

Type:	AssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAssignmentPolicyRule

You can pipe the AssignmentPolicyRule to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerAutoTagRule

March 11, 2024

Removes an AutoTagRule.

Syntax

```
Remove-BrokerAutoTagRule
       [-InputObject] <AutoTagRule[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerAutoTagRule
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Removes an AutoTagRule.

Examples

EXAMPLE 1

Removes RandomAllocatedCatalogs AutoTagRule.

```
Remove-BrokerAutoTagRule -Name RandomAllocatedCatalogs
```

Parameters

-InputObject

Specifies the AutoTagRule to remove.

Type:	AutoTagRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the AutoTagRule to remove.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AutoTagRule

AutoTagRules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_AutoTagRule
- Get-BrokerAutoTagRule
- New-BrokerAutoTagRule
- Set-BrokerAutoTagRule
- Rename-BrokerAutoTagRule

Remove-BrokerAutoTagRuleMetadata

March 11, 2024

Deletes AutoTagRule Metadata from the AutoTagRule objects

Syntax

```
Remove-BrokerAutoTagRuleMetadata
       [-InputObject] <AutoTagRule[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerAutoTagRuleMetadata cmdlet deletes Metadata from the AutoTagRule objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the AutoTagRule whose instance is pointed by $obj-Uid

```
Remove-BrokerAutoTagRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the AutoTagRule in the site

```
Get-BrokerAutoTagRule | Remove-BrokerAutoTagRuleMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the AutoTagRule object’s instance whose Metadata is to be deleted.

Type:	AutoTagRule[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAutoTagRule

You can pipe the AutoTagRule to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerCatalog

March 11, 2024

Removes catalogs from the site.

Syntax

```
Remove-BrokerCatalog
       [-InputObject] <Catalog[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerCatalog
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Remove catalogs from the site.

In order to remove a catalog from a site, the catalog must not contain machines. To remove a machine from a catalog use the Remove-BrokerMachine cmdlet. Note: in order to remove a machine from a catalog, it must not belong to a desktop group.

Examples

EXAMPLE 1

These commands delete the catalog with the name "MyCatalog".

```
Remove-BrokerCatalog -Name "MyCatalog"
Remove-BrokerCatalog -InputObject (Get-BrokerCatalog -Name "MyCatalog")
```

EXAMPLE 2

This command deletes all catalogs with names beginning with "test".

```
Remove-BrokerCatalog -Name 'test*'
```

EXAMPLE 3

Remove all the Remote PC catalogs that are associated with desktop group 42. Note that this only breaks the Remote PC relationships and does not delete the desktop groups.

```
Get-BrokerCatalog -RemotePCDesktopGroupUid 42 | Remove-BrokerCatalog -
    RemotePCDesktopGroup 42
```

Parameters

-InputObject

Specifies the catalog objects to delete.

Type:	Catalog[]
Position:	2
Default value:	Null

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the catalog to delete.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

You can pipe the catalogs to be deleted to Remove-BrokerCatalog.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_RemotePC
- New-BrokerCatalog
- Get-BrokerCatalog
- Rename-BrokerCatalog
- Set-BrokerCatalog
- New-BrokerDesktopGroup
- Remove-BrokerDesktopGroup
- Move-BrokerCatalog

Remove-BrokerCatalogMetadata

March 11, 2024

Deletes Catalog Metadata from the Catalog objects

Syntax

```
Remove-BrokerCatalogMetadata
       [-InputObject] <Catalog[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerCatalogMetadata cmdlet deletes Metadata from the Catalog objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the Catalog whose instance is pointed by $obj-Uid

```
Remove-BrokerCatalogMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the Catalog in the site

```
Get-BrokerCatalog | Remove-BrokerCatalogMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Catalog object's instance whose Metadata is to be deleted.

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerCatalog

You can pipe the Catalog to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerCatalogRebootSchedule

March 11, 2024

Removes the reboot schedule.

Syntax

```
Remove-BrokerCatalogRebootSchedule
       [-InputObject] <CatalogRebootSchedule[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerCatalogRebootSchedule
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Remove-BrokerCatalogRebootSchedule cmdlet is used to delete an existing catalog reboot schedule.

Examples

EXAMPLE 1

Deletes every catalog reboot schedule.

```
Get-BrokerCatalogRebootSchedule | Remove-BrokerCatalogRebootSchedule
```

EXAMPLE 2

Deletes the catalog reboot schedule having Uid 12.

```
Remove-BrokerCatalogRebootSchedule 12
```

EXAMPLE 3

Deletes the catalog reboot schedule named Accounting.

```
Remove-BrokerCatalogRebootSchedule -Name Accounting
```

Parameters

-InputObject

The catalog reboot schedule to be removed.

Type:	CatalogRebootSchedule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the catalog reboot schedule to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.CatalogRebootSchedule

Reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerCatalogRebootSchedule
- Set-BrokerCatalogRebootSchedule
- New-BrokerCatalogRebootSchedule
- Rename-BrokerCatalogRebootSchedule
- Stop-BrokerRebootCycle

Remove-BrokerConfigurationSlot

March 11, 2024

Removes a configuration slot.

Syntax

```
Remove-BrokerConfigurationSlot
       [-InputObject] <ConfigurationSlot[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerConfigurationSlot
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Removes a configuration slot from the site.

All machine configurations associated with this slot are also removed.

Examples

EXAMPLE 1

Remove the configuration slot named “User Profile Manager”.

```
1  Remove-BrokerConfigurationSlot -Name "User Profile Manager"
```

Parameters

-InputObject

Configuration slot to remove.

Type:	ConfigurationSlot[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Name of configuration slot to remove.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Configuration slot to remove

Configuration slots may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-BrokerConfigurationSlot
- Get-BrokerConfigurationSlot
- about_Broker_ConfigurationSlots

Remove-BrokerConfigurationSlotMetadata

March 11, 2024

Deletes ConfigurationSlot Metadata from the ConfigurationSlot objects

Syntax

```
Remove-BrokerConfigurationSlotMetadata
      [-InputObject] <ConfigurationSlot[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerConfigurationSlotMetadata cmdlet deletes Metadata from the ConfigurationSlot objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the ConfigurationSlot whose instance is pointed by $obj-Uid


```
Remove-BrokerConfigurationSlotMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the ConfigurationSlot in the site

```
Get-BrokerConfigurationSlot | Remove-BrokerConfigurationSlotMetadata -
    Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the ConfigurationSlot object’s instance whose Metadata is to be deleted.

Type:	ConfigurationSlot[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerConfigurationSlot

You can pipe the ConfigurationSlot to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerConfiguredFTA

March 11, 2024

Deletes one or more configured file type associations.

Syntax

```
Remove-BrokerConfiguredFTA
       [-InputObject] <ConfiguredFTA[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Deletes one or more file type associations configured for a published application. At least one configured file type association object must be specified.

Examples

EXAMPLE 1

Deletes all configured file type associations with an extension name of ".txt".

```
$ftas = Get-BrokerConfiguredFTA -ExtensionName ".txt"
Remove-BrokerConfiguredFTA $ftas
```

Parameters

-InputObject

Specifies the ConfiguredFTA objects to delete.

Type:	ConfiguredFTA[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ConfiguredFTA[]

One or more ConfiguredFTA objects can be supplied as input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerConfiguredFTA
- New-BrokerConfiguredFTA

Remove-BrokerControllerMetadata

March 11, 2024

Deletes Controller Metadata from the Controller objects

Syntax

```
Remove-BrokerControllerMetadata
      [-InputObject] <Controller[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerControllerMetadata cmdlet deletes Metadata from the Controller objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Controller whose instance is pointed by $obj-Uid

```
Remove-BrokerControllerMetadata -InputObject $obj-Uid -Name "
   MyMetadataName"
```


EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Controller in the site

```
Get-BrokerController | Remove-BrokerControllerMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the Controller object’s instance whose Metadata is to be deleted.

Type:	Controller[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerController

You can pipe the Controller to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerDelayedHostingPowerAction

March 11, 2024

Cancels one or more delayed power actions.

Syntax

```
Remove-BrokerDelayedHostingPowerAction
      [-InputObject] <DelayedHostingPowerAction[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerDelayedHostingPowerAction
      [-MachineName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Removes one or more delayed power actions that have not yet been queued for execution.

Examples

EXAMPLE 1

Cancels any pending delayed power actions for the machine called XD_VDA1.

```
Remove-BrokerHostingPowerAction -MachineName 'XD_VDA1'
```

Parameters

-InputObject

The power action to be cancelled.

Type:	DelayedHostingPowerAction[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Cancels only actions for machines whose name (of the form domain\machine) matches the specified string.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DelayedHostingPowerAction

The power action to be cancelled.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_PowerManagement
- Get-BrokerDelayedHostingPowerAction
- New-BrokerDelayedHostingPowerAction

Remove-BrokerDesktopGroup

March 11, 2024

Remove desktop groups from the system or remove them from a Remote PC catalog.

Syntax

```
Remove-BrokerDesktopGroup
      [-InputObject] <DesktopGroup[]>
      [-Force]
      [-RemotePCCatalog <Catalog>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerDesktopGroup
      [-Force]
      [-Name] <String>
      [-RemotePCCatalog <Catalog>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet has 2 functions:

- Remove desktop groups from the system.
- Break Remote PC associations between desktop groups and a catalog.

The Remote PC relationships are used by Remote PC automation to determine which desktop groups a machine in a particular Remote PC catalog can be published to. The assignment policy rules belonging to those desktop groups also determines the set of users that are allowed to be assigned to machines from the catalog.

Examples

EXAMPLE 1

Remove all desktop groups with names starting with “EMEA”.

```
Remove-BrokerDesktopGroup EMEA*
```

EXAMPLE 2

Remove all desktops that are currently disabled even if there are active sessions.

```
Get-BrokerDesktopGroup -Enabled $false | Remove-BrokerDesktopGroup -
    Force
```


EXAMPLE 3

Remove all the Remote PC desktop groups that are associated with catalog 42. Note that this only breaks the Remote PC relationships and does not delete the desktop groups.

```
Get-BrokerDesktopGroup -RemotePCCatalogUid 42 | Remove-
    BrokerDesktopGroup -RemotePCCatalog 42
```

Parameters

-InputObject

Specifies the desktop groups to remove.

Type:	DesktopGroup[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the desktop group to remove.

Type:	String
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Force

Remove desktop groups even if there are active sessions.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemotePCCatalog

When this parameter is specified, Remote PC desktop groups are removed from the specified Remote PC catalog.

Type:	Catalog
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

You can pipe desktop groups to Remove-BrokerDesktopGroup.

Outputs

None

By default, this cmdlet returns no output.

Notes

If a desktop group contains desktops when it is removed, these desktops are also removed (but the underlying broker machine remains).

A desktop group that still has active sessions cannot be removed unless the -Force switch is used.

Related Links

- about_Broker_Desktops
- about_Broker_RemotePC
- Get-BrokerDesktopGroup
- New-BrokerDesktopGroup
- Set-BrokerDesktopGroup
- Add-BrokerDesktopGroup
- Rename-BrokerDesktopGroup

- Move-BrokerDesktopGroup
- New-BrokerCatalog
- Remove-BrokerCatalog

Remove-BrokerDesktopGroupMetadata

March 11, 2024

Deletes DesktopGroup Metadata from the DesktopGroup objects

Syntax

```
Remove-BrokerDesktopGroupMetadata
       [-InputObject] <DesktopGroup[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerDesktopGroupMetadata cmdlet deletes Metadata from the DesktopGroup objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the DesktopGroup whose instance is pointed by $obj-Uid

```
Remove-BrokerDesktopGroupMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the DesktopGroup in the site


```
1 Get-BrokerDesktopGroup | Remove-BrokerDesktopGroupMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the DesktopGroup object's instance whose Metadata is to be deleted.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerDesktopGroup

You can pipe the DesktopGroup to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerDesktopGroupWebhook

March 11, 2024

Remove the webhook from a desktop group

Syntax

```
Remove-BrokerDesktopGroupWebhook
      [-InputObject] <DesktopGroupWebhook[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet is used to remove the configured webhook from a desktop group

Examples

EXAMPLE 1

Removes the webhook configured for the desktop group with Uid 1

```
Get-BrokerDesktopGroupWebhook -DesktopGroupUid 1 | Remove-
   BrokerDesktopGroupWebhook
```

Parameters

-InputObject

Sepcified the webhook to remove

Type:	DesktopGroupWebhook[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroupWebhook

You can pipe webhooks to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerDesktopGroup

Remove-BrokerEntitlementPolicyRule

March 11, 2024

Deletes a desktop rule from the site's entitlement policy.

Syntax

```
Remove-BrokerEntitlementPolicyRule
      [-InputObject] <EntitlementPolicyRule[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerEntitlementPolicyRule
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerEntitlementPolicyRule cmdlet deletes a desktop rule from the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

Deleting a rule does not affect existing sessions launched using the rule, but users cannot reconnect to those sessions if they are subsequently disconnected.

Examples

EXAMPLE 1

Deletes the desktop rule called Temp Workers from the entitlement policy. Existing desktop sessions launched using the rule are not affected, but users cannot reconnect to sessions if they are subsequently disconnected.


```
Remove-BrokerEntitlementPolicyRule 'Temp Workers'
```

EXAMPLE 2

Deletes all desktop rules from the entitlement policy applying to the Customer Support desktop group. This effectively removes all access to the desktops published from this group. Existing desktop sessions are not affected, but users cannot reconnect to sessions if they are subsequently disconnected.

```
$dg = Get-BrokerDesktopGroup 'Customer Support'
Get-BrokerEntitlementPolicyRule -DesktopGroupUid $dg.Uid | Remove-
    BrokerEntitlementPolicyRule
```

Parameters

-InputObject

The desktop rule to be deleted from the entitlement policy.

Type:	EntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the desktop rule to be deleted from the entitlement policy.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

The desktop rule to be deleted from the entitlement policy.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_ErrorHandling
- New-BrokerEntitlementPolicyRule
- Get-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule

Remove-BrokerEntitlementPolicyRuleMetadata

March 11, 2024

Deletes EntitlementPolicyRule Metadata from the EntitlementPolicyRule objects

Syntax

```
Remove-BrokerEntitlementPolicyRuleMetadata
      [-InputObject] <EntitlementPolicyRule[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerEntitlementPolicyRuleMetadata cmdlet deletes Metadata from the Entitlement-PolicyRule objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the EntitlementPolicyRule whose instance is pointed by $obj-Uid

```
Remove-BrokerEntitlementPolicyRuleMetadata -InputObject $obj-Uid -Name
    "MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the EntitlementPolicyRule in the site

```
Get-BrokerEntitlementPolicyRule | Remove-
    BrokerEntitlementPolicyRuleMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the EntitlementPolicyRule object's instance whose Metadata is to be deleted.

Type:	EntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule

You can pipe the EntitlementPolicyRule to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerGpoFilter

March 11, 2024

Removes a GPO filter.

Syntax

```
Remove-BrokerGpoFilter
       [-InputObject] <GpoFilter[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Remove a GPO filter.

Examples

EXAMPLE 1

Remove the filter with the specified ID.

```
Remove-BrokerGpoFilter ([Guid]"12345678-...")
```

Parameters

-InputObject

Specifies the GPO filter object to remove.

Type:	GpoFilter[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoFilter

GPO filters may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoFilter
- New-BrokerGpoFilter
- Set-BrokerGpoFilter

Remove-BrokerGpoPolicy

March 11, 2024

Remove a GPO policy.

Syntax

```
Remove-BrokerGpoPolicy
       [-InputObject] <GpoPolicy[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerGpoPolicy
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Remove a GPO policy.

Examples

EXAMPLE 1

Remove all policies that have the word 'test'in their names.

```
Get-BrokerGpoPolicy -Name '*test*' | Remove-BrokerGpoPolicy
```

Parameters

-InputObject

Specifies the GPO policy object to remove.

Type:	GpoPolicy[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the GPO policy object to remove.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoPolicy

You can pipe GPO policies to Remove-BrokerGpoPolicy.

Outputs

None

By default, this cmdlet returns no output.

Notes

Removing a policy changes the priorities of other existing policies.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoPolicy
- New-BrokerGpoPolicy
- Set-BrokerGpoPolicy

Remove-BrokerGpoPolicySet

March 11, 2024

Remove a GPO policy set.

Syntax

```
Remove-BrokerGpoPolicySet
      [-InputObject] <GpoPolicySet[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerGpoPolicySet
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Remove a GPO policy set. The policies, filters, and settings that are in the policy set are removed, as well as the policy set object itself.

Examples

EXAMPLE 1

Remove the specified policy set in the database.

```
Remove-BrokerGpoPolicySet ([Guid]"12345678-...")
```

Parameters

-InputObject

Specifies the policy set to remove.

Type:	GpoPolicySet[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the policy set to remove.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Broker.Admin.SDK.GpoPolicySet

Remove-BrokerGpoPolicySet removes a GPO policy set object.

Related Links

- about_Broker_GroupPolicy

- New-BrokerGpoPolicySet
- Get-BrokerGpoPolicySet
- Set-BrokerGpoPolicySet

Remove-BrokerGpoSetting

March 11, 2024

Remove a GPO setting.

Syntax

```
Remove-BrokerGpoSetting
      [-InputObject] <GpoSetting[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerGpoSetting
      [-SettingName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Remove a GPO setting.

Examples

EXAMPLE 1

Remove the setting with the specified GUID.

```
Remove-BrokerGpoSetting ([Guid]"12345678-...")
```

Parameters

-InputObject

Specifies the GPO setting object to remove.

Type:	GpoSetting[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SettingName

Specifies the name of the GPO setting object to remove.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoSetting

GPO settings may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoSetting
- New-BrokerGpoSetting
- Set-BrokerGpoSetting

Remove-BrokerHostingPowerAction

March 11, 2024

Cancel one or more pending power actions.

Syntax

```
Remove-BrokerHostingPowerAction
      [-InputObject] <HostingPowerAction[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerHostingPowerAction
      [-MachineName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Causes one or more of the pending power actions in the queue to be marked as canceled. The affected power actions are not sent to the hypervisor for processing, and take no further part in the queuing activity.

Power actions cannot be canceled once they have started to be processed by the hypervisor.

Examples

EXAMPLE 1

Cancels any pending power actions for the machine called 'XD_VDA1'.

```
Remove-BrokerHostingPowerAction -MachineName 'XD_VDA1'
```

EXAMPLE 2

Cancels any pending power actions requested in the last five minutes.

```
Get-BrokerHostingPowerAction -Filter {
  State -eq "Pending" -and RequestTime -gt "-00:05"  }
  | Remove-BrokerHostingPowerAction
```

Parameters

-InputObject

The power action to be canceled.

Type:	HostingPowerAction[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Cancels only actions for machines whose name (of the form domain\machine) matches the specified string.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.HostingPowerAction

The power action to be canceled.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_PowerManagement
- Get-BrokerHostingPowerAction
- New-BrokerHostingPowerAction
- Set-BrokerHostingPowerAction

Remove-BrokerHostingPowerActionMetadata

March 11, 2024

Deletes HostingPowerAction Metadata from the HostingPowerAction objects

Syntax

```
Remove-BrokerHostingPowerActionMetadata
      [-InputObject] <HostingPowerAction[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerHostingPowerActionMetadata cmdlet deletes Metadata from the HostingPowerAction objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the HostingPowerAction whose instance is pointed by $obj-Uid

```
Remove-BrokerHostingPowerActionMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the HostingPowerAction in the site

```
Get-BrokerHostingPowerAction | Remove-BrokerHostingPowerActionMetadata
    -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the HostingPowerAction object’s instance whose Metadata is to be deleted.

Type:	HostingPowerAction[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHostingPowerAction

You can pipe the HostingPowerAction to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerHypervisorAlertMetadata

March 11, 2024

Deletes HypervisorAlert Metadata from the HypervisorAlert objects

Syntax

```
Remove-BrokerHypervisorAlertMetadata
      [-InputObject] <HypervisorAlert[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

The Remove-BrokerHypervisorAlertMetadata cmdlet deletes Metadata from the HypervisorAlert objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the HypervisorAlert whose instance is pointed by $obj-Uid

```
Remove-BrokerHypervisorAlertMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the HypervisorAlert in the site

```
Get-BrokerHypervisorAlert | Remove-BrokerHypervisorAlertMetadata -Name
    "MyMetadataName"
```

Parameters

-InputObject

Specifies the HypervisorAlert object's instance whose Metadata is to be deleted.

Type:	HypervisorAlert[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHypervisorAlert

You can pipe the HypervisorAlert to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerHypervisorConnection

March 11, 2024

Removes a hypervisor connection from the system.

Syntax

```
Remove-BrokerHypervisorConnection
      [-InputObject] <HypervisorConnection[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerHypervisorConnection
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Remove-BrokerHypervisorConnection removes a hypervisor connection from the system. A hypervisor connection cannot be removed if it's being used by a machine.

Examples

EXAMPLE 1

This command removes a hypervisor connection by name.

```
Remove-BrokerHypervisorConnection -Name "Xen Server Connection"
```

EXAMPLE 2

Gets a hypervisor connection by preferred controller and removes it.

```
$hvConn = Get-BrokerHypervisorConnection -PreferredController "
    controllerName" -Name "Xen Server Connection"
Remove-BrokerHypervisorConnection -InputObject $hvConn
```

Parameters

-InputObject

Specifies the hypervisor connection object to remove.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the hypervisor connection object to remove.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.HypervisorConnection

You can pipe the hypervisor connection to be removed to Remove-BrokerHypervisorConnection.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerHypervisorConnection
- Set-BrokerHypervisorConnection
- New-BrokerHypervisorConnection

Remove-BrokerHypervisorConnectionMetadata

March 11, 2024

Deletes HypervisorConnection Metadata from the HypervisorConnection objects

Syntax

```
Remove-BrokerHypervisorConnectionMetadata
      [-InputObject] <HypervisorConnection[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerHypervisorConnectionMetadata cmdlet deletes Metadata from the Hypervisor-Connection objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the HypervisorConnection whose instance is pointed by $obj-Uid


```
Remove-BrokerHypervisorConnectionMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the HypervisorConnection in the site

```
Get-BrokerHypervisorConnection | Remove-
    BrokerHypervisorConnectionMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the HypervisorConnection object’s instance whose Metadata is to be deleted.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHypervisorConnection

You can pipe the HypervisorConnection to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerIcon

March 11, 2024

Remove an icon.

Syntax

```
Remove-BrokerIcon
       [-InputObject] <Icon[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Removes an icon from the database.

Examples

EXAMPLE 1

Removes the icon with Uid 3.

```
Remove-BrokerIcon 3
```

Parameters

-InputObject

Specifies the icon to remove.

Type:	Icon[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Icon

The icon to be removed can be piped into the cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

Note that if the icon is currently in use, for example, as a desktop icon, it cannot be removed until the association is cleared.

Related Links

- Get-BrokerIcon
- New-BrokerIcon
- Set-BrokerIconMetadata

Remove-BrokerIconMetadata

March 11, 2024

Deletes Icon Metadata from the Icon objects

Syntax

```
Remove-BrokerIconMetadata
      [-InputObject] <Icon[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerIconMetadata cmdlet deletes Metadata from the Icon objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Icon whose instance is pointed by $obj-Uid

```
Remove-BrokerIconMetadata -InputObject $obj-Uid -Name "MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Icon in the site

```
Get-BrokerIcon | Remove-BrokerIconMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Icon object’s instance whose Metadata is to be deleted.

Type:	Icon[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerIcon

You can pipe the Icon to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerImportDb

March 11, 2024

This cmdlet is for internal use only

Syntax

```
Remove-BrokerImportDb
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerImportedFTA

March 11, 2024

Deletes one or more imported file type associations.

Syntax

```
Remove-BrokerImportedFTA
      -DesktopGroupUids <Int32[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Deletes all of the imported file type associations belonging to one or more desktop groups. At least one desktop group must be specified.

Imported file type associations are grouped together based on the desktop group of the machine from which they were imported. All file types for a desktop group are deleted. There is no mechanism for deleting a subset imported file type associations for a specific desktop group.

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Examples

EXAMPLE 1

Deletes all imported file type associations

belonging to the “Sales VMs”desktop group.

```
$dg = Get-BrokerDesktopGroup -Name "Sales VMs"
Remove-BrokerImportedFTA -DesktopGroupUids $dg.Uid
```

Parameters

-DesktopGroupUids

Deletes the imported file type associations belonging

to specified desktop groups.

Type:	Int32[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Int32[]

An array of Uids for the desktop groups can be supplied as input. The desktop groups must be of the Private or Shared desktop kind.

Outputs

None

By default, this cmdlet returns no output.

Notes

If an imported file type association is used to create a new configured file type association and the imported file type association is subsequently deleted, the configured file type association is not affected.

Related Links

- about_Broker_Applications
- Get-BrokerImportedFTA
- Update-BrokerImportedFTA

Remove-BrokerLease

March 11, 2024

Remove the specified lease in the Database.

Syntax

```
Remove-BrokerLease
      [-InputObject] <Lease[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerLease
      [-Key] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Marks the specified lease for deletion. Note that the lease is eventually fully deleted when enough time has been allowed for the deletion to propagate to all controller machines in the site, but is immediately removed from lease search results.

Examples

EXAMPLE 1

Marks the specified lease for deletion.

```
$lease = Get-BrokerLease -Uid 1
Remove-BrokerLease $lease
```


Parameters

-InputObject

Specifies the lease to remove.

Type:	Lease[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Key

Specifies the lease key of the lease to remove. A pattern can be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Lease

The lease to be removed can be piped into the cmdlet.

Outputs

None

This cmdlet does not return any output.

Notes

The lease is marked for deletion after this cmdlet is run. Note that the lease is eventually fully deleted when enough time has been allowed for the deletion to propagate to all controller machines in the site, but is immediately removed from lease search results.

Related Links

- Update-BrokerLocalLeaseCache

Remove-BrokerLeaseMetadata

March 11, 2024

Deletes Lease Metadata from the Lease objects

Syntax

```
Remove-BrokerLeaseMetadata
      [-InputObject] <Lease[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerLeaseMetadata cmdlet deletes Metadata from the Lease objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Lease whose instance is pointed by $obj-Uid

```
Remove-BrokerLeaseMetadata -InputObject $obj-Uid -Name "MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Lease in the site

```
Get-BrokerLease | Remove-BrokerLeaseMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Lease object’s instance whose Metadata is to be deleted.

Type:	Lease[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerLease

You can pipe the Lease to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerLocalDb

March 11, 2024

This cmdlet is for internal use only

Syntax

```
Remove-BrokerLocalDb
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerMachine

March 11, 2024

Removes one or more machines from its desktop group or catalog.

Syntax

```
Remove-BrokerMachine
      [-InputObject] <Machine[]>
      [-Force]
      [-DesktopGroup <DesktopGroup>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerMachine
      [-Force]
      [-MachineName] <String>
      [-DesktopGroup <DesktopGroup>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerMachine cmdlet removes one or more machines from their desktop group or catalog. There are three forms:

- Use the -InputObject parameter to remove a single machine instance or

array of instances from their desktop group or catalog.

- Use the -MachineName parameter to remove the single named machine from

its group or catalog.

- Use pipelining to pipe machine instances to the command.

To remove machines from their desktop group use the -DesktopGroup parameter; the specified group must be the one that contains the machines. If more than one machine is being removed from its group they must all be members of the same group.

If the -DesktopGroup parameter is not used then the machines are removed from their catalog. Removing a machine from its catalog deletes the record of the machine from the Citrix Broker Service.

Machines cannot be removed from their catalog while they are members of a desktop group.

Examples

EXAMPLE 1

These all remove a single machine from a desktop group, identifying the group by instance, UID, or name.


```
Remove-BrokerMachine -InputObject $machine -DesktopGroup $desktopGroup
Remove-BrokerMachine -InputObject $machine -DesktopGroup 2
Remove-BrokerMachine $machine -DesktopGroup "MyDesktopGroup"
```

EXAMPLE 2

These remove the machine called “DOMAIN\MyMachine”from its desktop group.

```
Remove-BrokerMachine -MachineName "DOMAIN\MyMachine" -DesktopGroup 2
Remove-BrokerMachine DOMAIN\MyMachine -DesktopGroup "MyDesktopGroup"
Remove-BrokerMachine DOMAIN\MyMachine -DesktopGroup $desktopGroup
```

EXAMPLE 3

These all remove a machine from its catalog.

```
Remove-BrokerMachine -MachineName DOMAIN\MyMachine
Remove-BrokerMachine "DOMAIN\MyMachine"
Remove-BrokerMachine $machine
```

EXAMPLE 4

These find specific machines and remove them from their desktop group or catalog.

```
Get-BrokerMachine -Uid 3 | Remove-BrokerMachine -DesktopGroup $dg
Get-BrokerMachine -CatalogUid 4 | Remove-BrokerMachine
```

Parameters

-InputObject

An array of machines to be removed from their desktop group or catalog.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

The name of the single machine to remove (must match the MachineName property of the machine).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Force

Forces removal of machine from a desktop group even if it is still in use (that is, there are user sessions running on the machine). Forcing removal of a machine does not disconnect or logoff the user sessions.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroup

The desktop group from which the machines are to be removed, specified by name, UID, or instance.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

You can pipe in the machines to be removed.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Machines
- Add-BrokerMachine
- Get-BrokerMachine

Remove-BrokerMachineCommand

March 11, 2024

Cancel a pending command queued for delivery to a desktop.

Syntax

```
Remove-BrokerMachineCommand
      [-InputObject] <MachineCommand[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Sets the state of a pending command queued for delivery to a desktop to Canceled. The command is not removed from the system.

Examples

EXAMPLE 1

Cancel all pending commands.

```
Get-BrokerMachineCommand | Remove-BrokerMachineCommand
```


EXAMPLE 2

Cancel all pending commands that have the category "UPM".

```
1  Get-BrokerMachineCommand -Category "UPM" | Remove-BrokerMachineCommand
```

Parameters

-InputObject

Commands to cancel.

Type:	MachineCommand[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.MachineCommand

Commands to cancel.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerMachineCommand
- New-BrokerMachineCommand

Remove-BrokerMachineCommandMetadata

March 11, 2024

Deletes MachineCommand Metadata from the MachineCommand objects

Syntax

```
Remove-BrokerMachineCommandMetadata
       [-InputObject] <MachineCommand[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerMachineCommandMetadata cmdlet deletes Metadata from the MachineCommand objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the MachineCommand whose instance is pointed by $obj-Uid

```
Remove-BrokerMachineCommandMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the MachineCommand in the site

```
Get-BrokerMachineCommand | Remove-BrokerMachineCommandMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the MachineCommand object’s instance whose Metadata is to be deleted.

Type:	MachineCommand[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachineCommand

You can pipe the MachineCommand to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerMachineConfiguration

March 11, 2024

Deletes a machine configuration from the site or removes the association from a desktop group.

Syntax

```
Remove-BrokerMachineConfiguration
       [-InputObject] <MachineConfiguration[]>
       [-Application <Application>]
       [-DesktopGroup <DesktopGroup>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-BrokerMachineConfiguration
      [-Name] <String>
      [-Application <Application>]
      [-DesktopGroup <DesktopGroup>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet has three functions:

- Break associations between desktop groups and machine configurations.
- Break associations between applications and machine configurations.
- Remove machine configurations from the system.

If no desktop group or application is specified, then the Remove-MachineConfiguration cmdlet removes the machine configuration from the site.

If a desktop group is specified, then the Remove-MachineConfiguration cmdlet removes the association between the machine configuration and that desktop group. In this case, the machine configuration is not removed from the site.

If an application is specified, then the Remove-MachineConfiguration cmdlet removes the association between the machine configuration and that application. Again, in this case, the machine configuration is not removed from the site.

Examples

EXAMPLE 1

Removes the machine configuration named “UPM\Finance”.

```
Remove-BrokerMachineConfiguration -Name UPM\Finance
```

EXAMPLE 2

Removes the association of the machine configuration named “Receiver\HumanResources”from the “SharedWorkers”desktop group.

```
Remove-BrokerMachineConfiguration -Name Receiver\HumanResources -
    DesktopGroup SharedWorkers
```


Parameters

-InputObject

Machine configuration to remove.

Type:	MachineConfiguration[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Name of machine configuration for which the remove operation applies.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Application

The application from which this machine configuration is to be removed.

Type:	Application
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DesktopGroup

The desktop group from which this machine configuration is to be removed.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.MachineConfiguration

Machine configuration to remove

Outputs

None

By default, this cmdlet returns no output.

Notes

A machine configuration can only be removed if it is not currently applied to a desktop group.

Related Links

- New-BrokerMachineConfiguration
- Get-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

Remove-BrokerMachineConfigurationMetadata

March 11, 2024

Deletes MachineConfiguration Metadata from the MachineConfiguration objects

Syntax

```
Remove-BrokerMachineConfigurationMetadata
       [-InputObject] <MachineConfiguration[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerMachineConfigurationMetadata cmdlet deletes Metadata from the MachineConfiguration objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the MachineConfiguration whose instance is pointed by $obj-Uid

```
Remove-BrokerMachineConfigurationMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the MachineConfiguration in the site

```
Get-BrokerMachineConfiguration | Remove-
    BrokerMachineConfigurationMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the MachineConfiguration object's instance whose Metadata is to be deleted.

Type:	MachineConfiguration[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachineConfiguration

You can pipe the MachineConfiguration to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerMachineMetadata

March 11, 2024

Deletes Machine Metadata from the Machine objects

Syntax

```
Remove-BrokerMachineMetadata
       [-InputObject] <Machine[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Remove-BrokerMachineMetadata cmdlet deletes Metadata from the Machine objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Machine whose instance is pointed by $obj-Uid

```
Remove-BrokerMachineMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Machine in the site

```
Get-BrokerMachine | Remove-BrokerMachineMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Machine object’s instance whose Metadata is to be deleted.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachine

You can pipe the Machine to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerPowerTimeScheme

March 11, 2024

Deletes an existing power time scheme.

Syntax

```
Remove-BrokerPowerTimeScheme
      [-InputObject] <PowerTimeScheme[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerPowerTimeScheme
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerPowerTimeScheme cmdlet deletes a power time scheme from the system, and leaves the days that the time scheme used to cover for the associated desktop group as defaulting to all hours off-peak and all hours with pool size of -1.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

Examples

EXAMPLE 1

Deletes the power time scheme named 'Development Weekdays'.

```
Remove-BrokerPowerTimeScheme -Name 'Development Weekdays'
```

EXAMPLE 2

Deletes all power time schemes for the desktop group named 'Finance desk1'.

```
Get-BrokerPowerTimeScheme -DesktopGroupUid (Get-BrokerDesktopGroup -
    name 'Finance desk1').Uid  | Remove-BrokerPowerTimeScheme
```

Parameters

-InputObject

The power time scheme to be deleted.

Type:	PowerTimeScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the power time scheme to be deleted.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.PowerTimeScheme

The power time scheme to be deleted.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_PowerManagement
- Get-BrokerPowerTimeScheme
- Set-BrokerPowerTimeScheme
- New-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme

Remove-BrokerPowerTimeSchemeMetadata

March 11, 2024

Deletes PowerTimeScheme Metadata from the PowerTimeScheme objects

Syntax

```
Remove-BrokerPowerTimeSchemeMetadata
       [-InputObject] <PowerTimeScheme[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerPowerTimeSchemeMetadata cmdlet deletes Metadata from the PowerTimeScheme objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the PowerTimeScheme whose instance is pointed by $obj-Uid

```
Remove-BrokerPowerTimeSchemeMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the PowerTimeScheme in the site

```
Get-BrokerPowerTimeScheme | Remove-BrokerPowerTimeSchemeMetadata -Name
    "MyMetadataName"
```

Parameters

-InputObject

Specifies the PowerTimeScheme object’s instance whose Metadata is to be deleted.

Type:	PowerTimeScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerPowerTimeScheme

You can pipe the PowerTimeScheme to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerRebootCycleMetadata

March 11, 2024

Deletes RebootCycle Metadata from the RebootCycle objects

Syntax

```
Remove-BrokerRebootCycleMetadata
       [-InputObject] <RebootCycle[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerRebootCycleMetadata cmdlet deletes Metadata from the RebootCycle objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the RebootCycle whose instance is pointed by $obj-Uid

```
Remove-BrokerRebootCycleMetadata -InputObject $obj-Uid -Name "
   MyMetadataName"
```


EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the RebootCycle in the site

```
Get-BrokerRebootCycle | Remove-BrokerRebootCycleMetadata -Name "
    MyMetadataName"
```

Parameters

-InputObject

Specifies the RebootCycle object’s instance whose Metadata is to be deleted.

Type:	RebootCycle[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerRebootCycle

You can pipe the RebootCycle to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerRebootSchedule

March 11, 2024

Removes the reboot schedule.

Syntax

```
Remove-BrokerRebootSchedule
       [-InputObject] <RebootSchedule[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerRebootSchedule
       [-DesktopGroupName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerRebootSchedule cmdlet is used to delete an existing reboot schedule.

Examples

EXAMPLE 1

Deletes every reboot schedule.

```
Get-BrokerRebootSchedule | Remove-BrokerRebootSchedule
```

EXAMPLE 2

Deletes the reboot schedule for the desktop group having Uid 12.

```
Remove-BrokerRebootSchedule 12
```


EXAMPLE 3

Deletes the reboot schedule for the desktop group named Accounting.

```
1  Remove-BrokerRebootSchedule -DesktopGroupName Accounting
```

Parameters

-InputObject

The reboot schedule to be deleted.

Type:	RebootSchedule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose reboot schedule is to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootSchedule

Reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerRebootSchedule
- Set-BrokerRebootSchedule
- New-BrokerRebootSchedule
- Stop-BrokerRebootCycle

Remove-BrokerRebootScheduleV2

March 11, 2024

Removes the reboot schedule.

Syntax

```
Remove-BrokerRebootScheduleV2
      [-InputObject] <RebootScheduleV2[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerRebootScheduleV2
      [-Name] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerRebootScheduleV2 cmdlet is used to delete an existing reboot schedule.

Examples

EXAMPLE 1

Deletes every reboot schedule.

```
Get-BrokerRebootScheduleV2 | Remove-BrokerRebootScheduleV2
```

EXAMPLE 2

Deletes the reboot schedule having Uid 12.

```
Remove-BrokerRebootScheduleV2 12
```

EXAMPLE 3

Deletes the reboot schedule named Accounting.


```
Remove-BrokerRebootScheduleV2 -Name Accounting
```

Parameters

-InputObject

The reboot schedule to be removed.

Type:	RebootScheduleV2[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the reboot schedule to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootScheduleV2

Reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerRebootScheduleV2
- Set-BrokerRebootScheduleV2
- New-BrokerRebootScheduleV2

- Rename-BrokerRebootScheduleV2
- Stop-BrokerRebootCycle

Remove-BrokerRebootScheduleV2Metadata

March 11, 2024

Deletes RebootScheduleV2 Metadata from the RebootScheduleV2 objects

Syntax

```
Remove-BrokerRebootScheduleV2Metadata
       [-InputObject] <RebootScheduleV2[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerRebootScheduleV2Metadata cmdlet deletes Metadata from the RebootScheduleV2 objects.

Examples

EXAMPLE 1

This command deletes the Metadata "MyMetadataName"key-value pair for the RebootScheduleV2 whose instance is pointed by $obj-Uid

```
Remove-BrokerRebootScheduleV2Metadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata "MyMetadataName"key-value pair for all the RebootScheduleV2 in the site

```
Get-BrokerRebootScheduleV2 | Remove-BrokerRebootScheduleV2Metadata -
    Name "MyMetadataName"
```


Parameters

-InputObject

Specifies the RebootScheduleV2 object's instance whose Metadata is to be deleted.

Type:	RebootScheduleV2[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerRebootScheduleV2

You can pipe the RebootScheduleV2 to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerRemotePCAccount

March 11, 2024

Delete RemotePCAccounts from the system.

Syntax

```
Remove-BrokerRemotePCAccount
      [-InputObject] <RemotePCAccount[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Delete RemotePCAccounts from the site.

Examples

EXAMPLE 1

Delete RemotePCAccount 42.

```
Remove-BrokerRemotePCAccount 42
```

EXAMPLE 2

Delete the 'any'OU RemotePCAccount.

```
Get-BrokerRemotePCAccount -OU 'any' | Remove-BrokerRemotePCAccount
```

EXAMPLE 3

Delete all RemotePCAccounts.

```
Get-BrokerRemotePCAccount | Remove-BrokerRemotePCAccount
```

Parameters

-InputObject

Specifies the RemotePCAccounts to delete.

Type:	RemotePCAccount[]
Position:	2

Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RemotePCAccount

You can pipe the RemotePCAccounts to be deleted into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_RemotePC
- Get-BrokerRemotePCAccount
- New-BrokerRemotePCAccount
- Set-BrokerRemotePCAccount

Remove-BrokerScope

March 11, 2024

Remove the specified catalog/desktop group from the given scope(s).

Syntax

```
Remove-BrokerScope
      [-InputObject] <Scope[]>
      [-ApplicationGroup <ApplicationGroup>]
      [-Catalog <Catalog>]
      [-DesktopGroup <DesktopGroup>]
      [-PolicySet <GpoPolicySet>]
      [-Tag <Tag>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerScope cmdlet is used to remove a scopeable object object from the given scope(s). A scopeable object is one of:

- a catalog
- a desktop group
- an application group

To remove a scopeable object from a scope you need permission to change the scopes of the scopeable object.

If the scopeable object is not in a specified scope, that scope will be silently ignored.

Examples

EXAMPLE 1

Removes the “Win7 Desktops”desktop group from both the Sales and Marketing scopes.

```
Remove-BrokerScope Sales,Marketing -DesktopGroup "Win7 Desktops"
```

EXAMPLE 2

Removes the “Win7 Desktops”desktop group from both the Sales and Marketing scopes.

```
Sales,Marketing | Remove-BrokerScope -DesktopGroup 'Win7 Desktops'
```

Parameters

-InputObject

Specifies the scopes to remove the object from.

Type:	Scope[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ApplicationGroup

Specifies the application group object to be removed.

Type:	ApplicationGroup

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Catalog

Specifies the catalog object to be removed.

Type:	Catalog
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Specifies the desktop group object to be removed.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PolicySet

Specifies the policy set object to be removed.

Type:	GpoPolicySet
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Tag

Specifies the tag object to be removed.

Type:	Tag
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Scope

You can pipe scopes to Remove-BrokerScope.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerSessionLinger

March 11, 2024

Removes a session linger setting.

Syntax

```
Remove-BrokerSessionLinger
       [-InputObject] <SessionLinger[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-BrokerSessionLinger
      [-DesktopGroupName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerSessionLinger cmdlet is used to delete an existing session linger setting.

Examples

EXAMPLE 1

Deletes every session linger setting for all desktop groups.

```
Get-BrokerSessionLinger | Remove-BrokerSessionLinger
```

Parameters

-InputObject

The session linger setting to be deleted.

Type:	SessionLinger[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose session linger setting is to be removed.

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.SessionLinger

Session linger settings may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-BrokerSessionLinger
- Get-BrokerSessionLinger
- Set-BrokerSessionLinger

Remove-BrokerSessionMetadata

March 11, 2024

Deletes Session Metadata from the Session objects

Syntax

```
Remove-BrokerSessionMetadata
      [-InputObject] <Session[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerSessionMetadata cmdlet deletes Metadata from the Session objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Session whose instance is pointed by $obj-Uid

```
Remove-BrokerSessionMetadata -InputObject $obj-Uid -Name "
    MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Session in the site

```
Get-BrokerSession | Remove-BrokerSessionMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Session object’s instance whose Metadata is to be deleted.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerSession

You can pipe the Session to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerSessionPreLaunch

March 11, 2024

Removes a session pre-launch setting.

Syntax

```
Remove-BrokerSessionPreLaunch
      [-InputObject] <SessionPreLaunch[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-BrokerSessionPreLaunch
      [-DesktopGroupName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerSessionPreLaunch cmdlet is used to delete an existing session pre-launch setting.

Examples

EXAMPLE 1

Deletes every session pre-launch setting for all desktop groups.

```
Get-BrokerSessionPreLaunch | Remove-BrokerSessionPreLaunch
```


Parameters

-InputObject

The session pre-launch setting to be deleted.

Type:	SessionPreLaunch[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose session pre-launch setting is to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.SessionPreLaunch

Session pre-launch settings may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-BrokerSessionPreLaunch
- Get-BrokerSessionPreLaunch
- Set-BrokerSessionPreLaunch

Remove-BrokerSiteMetadata

March 11, 2024

Deletes Site Metadata from the Site objects

Syntax

```
Remove-BrokerSiteMetadata
      [[-InputObject] <Site[]>]
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-BrokerSiteMetadata cmdlet deletes Metadata from the Site objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Site

```
Remove-BrokerSiteMetadata -Name "MyMetadataName"
```

Parameters

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-InputObject

Specifies the Site object's instance whose Metadata is to be deleted.

Type:	Site[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerSite

You can pipe the Site to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerTag

March 11, 2024

Removes tag to object associations or deletes tags from the site altogether.

Syntax

```
Remove-BrokerTag
       [-InputObject] <Tag[]>
       [-Force]
       [-Application <Application>]
       [-ApplicationGroup <ApplicationGroup>]
       [-Catalog <Catalog>]
       [-Desktop <Desktop>]
       [-DesktopGroup <DesktopGroup>]
       [-Machine <Machine>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-BrokerTag
       [-Tags] <Tag[]>
       [-AllApplications]
       [-AllApplicationGroups]
       [-AllDesktops]
       [-AllDesktopGroups]
       [-AllCatalogs]
       [-AllMachines]
       [-AllObjects]
       [-Force]
       [-Application <Application>]
       [-ApplicationGroup <ApplicationGroup>]
       [-Catalog <Catalog>]
       [-Desktop <Desktop>]
       [-DesktopGroup <DesktopGroup>]
       [-Machine <Machine>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerTag
       [-Force]
       [-Name] <String>
       [-Application <Application>]
       [-ApplicationGroup <ApplicationGroup>]
       [-Catalog <Catalog>]
       [-Desktop <Desktop>]
       [-DesktopGroup <DesktopGroup>]
       [-Machine <Machine>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Removes the association between tags and objects within the site, or deletes tags from the site altogether.

To remove an association, supply one of the Application, Machine, Desktop or DesktopGroup parameters.

To delete a tag entirely, together with any associations between the tag and other objects in the site, specify the tag without any associated object parameter.

Examples

EXAMPLE 1

Removes the association between a tag and a desktop. The tag itself continues to exist in the site.

```
Remove-BrokerTag $tag -Machine $machine
```

EXAMPLE 2

Deletes the tag from the site also removing any associations that may exist between the tag and other objects.

```
Remove-BrokerTag $tag
```

Parameters

-Tags

Specifies one or more tag objects.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies one or more tag objects.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies a tag by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-AllApplications

Remove the specified tags from all applications.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllApplicationGroups

Remove the specified tags from all application groups.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllDesktops

Remove the specified tags from all desktops.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllDesktopGroups

Remove the specified tags from all desktop groups.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllCatalogs

Remove the specified tags from all catalogs.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllMachines

Remove the specified tags from all machines.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllObjects

Remove the specified tags from all objects.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

Remove tags even if they are used by group policy.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Application

Removes the association between the given tag and application.

Type:	Application
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ApplicationGroup

Removes the association between the given tag and application group.

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Catalog

Removes the association between the given tag and the catalog group.

Type:	Catalog
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Desktop

Removes the association between the given tag and desktop.

Type:	Desktop
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroup

Removes the association between the given tag and desktop group.

Type:	DesktopGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Machine

Removes the association between the given tag and machine.

Type:	Machine
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Tag

Tags may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Add-BrokerTag
- Get-BrokerTag
- New-BrokerTag
- Rename-BrokerTag
- Set-BrokerTag

Remove-BrokerTagMetadata

March 11, 2024

Deletes Tag Metadata from the Tag objects

Syntax

```
Remove-BrokerTagMetadata
       [-InputObject] <Tag[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerTagMetadata cmdlet deletes Metadata from the Tag objects.

Examples

EXAMPLE 1

This command deletes the Metadata “MyMetadataName”key-value pair for the Tag whose instance is pointed by $obj-Uid

```
Remove-BrokerTagMetadata -InputObject $obj-Uid -Name "MyMetadataName"
```

EXAMPLE 2

This command deletes the Metadata “MyMetadataName”key-value pair for all the Tag in the site

```
Get-BrokerTag | Remove-BrokerTagMetadata -Name "MyMetadataName"
```

Parameters

-InputObject

Specifies the Tag object’s instance whose Metadata is to be deleted.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata to be deleted

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerTag

You can pipe the Tag to delete the metadata.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerUniversalClaim

March 11, 2024

Remove a Broker UniversalClaim mapping

Syntax

```
Remove-BrokerUniversalClaim
       [-InputObject] <UniversalClaim[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-BrokerUniversalClaim cmdlet removes a UniversalClaim mapping that was previously added. Mappings are added automatically, and should not normally need to be removed.

Examples

EXAMPLE 1

Removes a UniversalClaim mapping

```
Remove-BrokerUniversalClaim -InputObject $ClaimToRemove
```

Parameters

-InputObject

Specifies the objects to be removed

Type:	UniversalClaim[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.UniversalClaim

You can pipe UniversalClaims to be removed by Remove-BrokerUniversalClaim

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-BrokerUser

March 11, 2024

Remove broker user objects from another broker object

Syntax

```
Remove-BrokerUser
       [-InputObject] <User[]>
       [-ApplicationGroup <ApplicationGroup>]
       [-Application <Application>]
       [-SessionLinger <SessionLinger>]
       [-SessionPreLaunch <SessionPreLaunch>]
       [-Machine <Machine>]
       [-PrivateDesktop <PrivateDesktop>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerUser
       [-Name] <String>
       [-ApplicationGroup <ApplicationGroup>]
       [-Application <Application>]
       [-SessionLinger <SessionLinger>]
       [-SessionPreLaunch <SessionPreLaunch>]
       [-Machine <Machine>]
       [-PrivateDesktop <PrivateDesktop>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
```



```
11          [<CommonParameters>]
```

Description

The Remove-BrokerUser cmdlet removes broker user objects from another specified object, such as a broker private desktop, to which the user had previously been added.

Examples

EXAMPLE 1

Remove the assignment of the specified private desktop to the specified user.

```
1  Remove-BrokerUser "DOMAIN\UserName" -PrivateDesktop "DOMAIN\MachineName
       "
```

Parameters

-InputObject

Specifies the user objects to remove.

Type:	User[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the user objects to remove, based on their Name property.

Type:	String
Position:	2
Default value:	Null

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ApplicationGroup

The application group from which to remove the user

Type:	ApplicationGroup
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Application

The application from which to remove the user

Type:	Application
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionLinger

The desktop group session linger setting from which to remove the user.

Type:	SessionLinger
Position:	Named

Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionPreLaunch

The desktop group session pre-launch setting from which to remove the user.

Type:	SessionPreLaunch
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Machine

The machine from which to remove the user

Type:	Machine
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PrivateDesktop

The desktop from which to remove the user

Type:	PrivateDesktop
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.USer

You can pipe the users to be removed to Remove-BrokerUser.

Outputs

None

By default, this cmdlet returns no output.

Notes

Specify one of the -Machine or -PrivateDesktop parameters only.

Related Links

- Add-BrokerUser
- Get-BrokerUser

Remove-BrokerUserZonePreference

March 11, 2024

Removes any zone preference associated with a user/group account in this site

Syntax

```
Remove-BrokerUserZonePreference
       [-InputObject] <UserZonePreference[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-BrokerUserZonePreference
       [-Name] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Remove-BrokerUserZonePreference cmdlet removes any preferred home zone associated with a user/group account.

Removing a home preference means that the account no longer has an influence on any choice of zone used for resource launches.

Examples

EXAMPLE 1

Removes any home zone preference associated with the APAC\sales account for this site.

```
Remove-BrokerUserZonePreference APAC\sales
```

Parameters

-InputObject

The account zone preference to be removed.

Type:	UserZonePreference[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the user/group account whose home zone preference is to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.UserZonePreference

The account zone preference to be removed.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Rename-BrokerAccessPolicyRule

March 11, 2024

Renames a rule in the site's access policy.

Syntax

```
Rename-BrokerAccessPolicyRule
      [-InputObject] <AccessPolicyRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerAccessPolicyRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Rename-BrokerAccessPolicyRule cmdlet renames a rule in the site's access policy. The Name property of the rule is changed.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Examples

EXAMPLE 1

Renames the access policy rule called Sales to TeleSales. The new name of the rule must be unique in the access policy.

```
1  Rename-BrokerAccessPolicyRule 'Sales' -NewName 'TeleSales'
```

Parameters

-InputObject

The access policy rule to be renamed.

Type:	AccessPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The existing name of the access policy rule to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name for the access policy rule being renamed. The new name must not match that of any other existing rules in the policy.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AccessPolicyRule

The access policy rule to be renamed.

Outputs

None or Citrix.Broker.Admin.SDK.AccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AccessPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- New-BrokerAccessPolicyRule
- Get-BrokerAccessPolicyRule
- Set-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule

Rename-BrokerAdminFolder

March 11, 2024

Renames a folder

Syntax

```
Rename-BrokerAdminFolder
       [-InputObject] <AdminFolder[]>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-BrokerAdminFolder
       [-Name] <String>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-BrokerAdminFolder cmdlet renames a folder for organising objects for administration purposes (for example, Applications).

The following special characters are not allowed in the new FolderName: \ / ; : # . * ? = < > | [ ] ( ) “‘

Examples

EXAMPLE 1

Renames the folder called XXX within the folder F1\ to YYY

```
Rename-BrokerAdminFolder F1\XXX\ YYY
```


Parameters

-InputObject

The folder(s) to be renamed

Type:	AdminFolder[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

A pattern matching the names of folders to be renamed

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The name the new folder(s) should have.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Depends on parameter

Parameters can be piped by property name.

Outputs

None or Citrix.Broker.Admin.SDK.AdminFolder

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AdminFolder object.

Related Links

- Get-BrokerAdminFolder
- New-BrokerAdminFolder

Rename-BrokerAppAssignmentPolicyRule

March 11, 2024

Renames an application rule in the site's assignment policy.

Syntax

```
Rename-BrokerAppAssignmentPolicyRule
      [-InputObject] <AppAssignmentPolicyRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Rename-BrokerAppAssignmentPolicyRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Rename-BrokerAppAssignmentPolicyRule cmdlet renames an application rule in the site's assignment policy. The Name property of the rule is changed.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

Examples

EXAMPLE 1

Renames the application rule in the assignment policy called Offshore to Remote Workers. The new name of the rule must be unique in the assignment policy.

```
Rename-BrokerAppAssignmentPolicyRule 'Offshore' -NewName 'Remote
   Workers'
```

Parameters

-InputObject

The application rule in the assignment policy to be renamed.

Type:	AppAssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The existing name of the application rule in the assignment policy to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name of the application rule in the assignment policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

The application rule in the assignment policy being renamed.

Outputs

None or Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppAssignmentPolicyRule
- Get-BrokerAppAssignmentPolicyRule
- Set-BrokerAppAssignmentPolicyRule
- Remove-BrokerAppAssignmentPolicyRule

Rename-BrokerAppEntitlementPolicyRule

March 11, 2024

Renames an application rule in the site's entitlement policy.

Syntax

```
Rename-BrokerAppEntitlementPolicyRule
      [-InputObject] <AppEntitlementPolicyRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerAppEntitlementPolicyRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

The Rename-BrokerAppEntitlementPolicyRule cmdlet renames an application rule in the site's entitlement policy. The Name property of the rule is changed.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

Examples

EXAMPLE 1

Renames the application rule in the entitlement policy called Prod Dev to Product Development. The new name of the rule must be unique in the entitlement policy.

```
1  Rename-BrokerAppEntitlementPolicyRule 'Prod Dev' -NewName 'Product
       Development'
```

Parameters

-InputObject

The application rule in the entitlement policy to be renamed.

Type:	AppEntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The existing name of the application rule in the entitlement policy to be renamed.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name of the application rule in the entitlement policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

The application rule in the entitlement policy being renamed.

Outputs

None or Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppEntitlementPolicyRule
- Get-BrokerAppEntitlementPolicyRule
- Set-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule

Rename-BrokerApplication

March 11, 2024

Renames an application.

Syntax

```
Rename-BrokerApplication
       [-InputObject] <Application[]>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-BrokerApplication
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-BrokerApplication cmdlet changes the administrative name of an application. An application cannot have the same name as another application.

Renaming an application does not alter its published name. To change the name with which this application appears to end-users, set a new value for the PublishedName property using the Set-BrokerApplication cmdlet.

Renaming an application does not alter its BrowserName. If the BrowserName property also needs to be changed, use the Set-BrokerApplication cmdlet to modify it.

Examples

EXAMPLE 1

Renames the application with name "Old Name"to "New Name".

```
Rename-BrokerApplication -Name "Old Name" -NewName "New Name"
```

EXAMPLE 2

Renames application with the Uid 1 to "New Name", showing the result.

```
Get-BrokerApplication -Uid 1 | Rename-BrokerApplication -NewName "New
    Name" -PassThru
```

Parameters

-InputObject

Specifies the application to rename.

Type:	Application[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application to rename.

Type:	String
Position:	2

Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name for the application.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Application

You can pipe applications to Rename-BrokerApplication.

Outputs

None or Citrix.Broker.Admin.SDK.Application

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Application object.

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Set-BrokerApplication
- Get-BrokerApplication
- Remove-BrokerApplication

Rename-BrokerApplicationGroup

March 11, 2024

Renames an application group.

Syntax

```
Rename-BrokerApplicationGroup
       [-InputObject] <ApplicationGroup[]>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-BrokerApplicationGroup
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-BrokerApplicationGroup cmdlet changes the name of an application group. An application group cannot have the same name as another application group.

Application group names are not visible to end users.

Examples

EXAMPLE 1

Renames the application group with name “Ofice”to “Office”.

```
Rename-BrokerApplicationGroup -Name "Ofice" -NewName "Office"
```

EXAMPLE 2

Renames application group with the Uid 1 to “New Name”, showing the result.

```
Get-BrokerApplicationGroup -Uid 1 | Rename-BrokerApplicationGroup -
    NewName "New Name" -PassThru
```

Parameters

-InputObject

Specifies the application group to rename.

Type:	ApplicationGroup[]
Position:	2
Default value:	Nul
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application group to rename.

Type:	String
Position:	2
Default value:	Null
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the application group.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ApplicationGroup

You can pipe application groups to Rename-BrokerApplicationGroup.

Outputs

None or Citrix.Broker.Admin.SDK.ApplicationGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.ApplicationGroup object.

Related Links

- about_Broker_Applications
- Add-BrokerApplicationGroup

- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Move-BrokerApplicationGroup
- Set-BrokerApplicationGroup

Rename-BrokerAssignmentPolicyRule

March 11, 2024

Renames a desktop rule in the site's assignment policy.

Syntax

```
Rename-BrokerAssignmentPolicyRule
      [-InputObject] <AssignmentPolicyRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerAssignmentPolicyRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Rename-BrokerAssignmentPolicyRule cmdlet renames a desktop rule in the site's assignment policy. The Name property of the rule is changed.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Examples

EXAMPLE 1

Renames the desktop rule in the assignment policy called Offshore to Remote Workers. The new name of the rule must be unique in the assignment policy.

```
Rename-BrokerAssignmentPolicyRule 'Offshore' -NewName 'Remote Workers'
```

Parameters

-InputObject

The desktop rule in the assignment policy to be renamed.

Type:	AssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The existing name of the desktop rule in the assignment policy to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name of the desktop rule in the assignment policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

The desktop rule in the assignment policy being renamed.

Outputs

None, or Citrix.Broker.Admin.SDK.AssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AssignmentPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_ErrorHandling
- New-BrokerAssignmentPolicyRule

- Get-BrokerAssignmentPolicyRule
- Set-BrokerAssignmentPolicyRule
- Remove-BrokerAssignmentPolicyRule

Rename-BrokerAutoTagRule

March 11, 2024

Renames the AutoTagRule.

Syntax

```
Rename-BrokerAutoTagRule
      [-InputObject] <AutoTagRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerAutoTagRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Renames the AutoTagRule.

Examples

EXAMPLE 1

Renames RandomAllocatedCatalogs to StaticAllocatedCatalogs.

```
Rename-BrokerAutoTagRule -Name RandomAllocatedCatalogs -NewName
    StaticAllocatedCatalogs
```


Parameters

-InputObject

Specifies the AutoTagRule object to rename.

Type:	AutoTagRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the AutoTagRule be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name for the AutoTagRule.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AutoTagRule

AutoTagRule may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_AutoTagRule
- Get-BrokerAutoTagRule
- New-BrokerAutoTagRule
- Set-BrokerAutoTagRule
- Remove-BrokerAutoTagRule

Rename-BrokerCatalog

March 11, 2024

Renames a catalog.

Syntax

```
Rename-BrokerCatalog
       [-InputObject] <Catalog[]>
       [-PassThru]
       [-NewName] <String>
```



```
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Rename-BrokerCatalog
        [-Name] <String>
        [-PassThru]
        [-NewName] <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

The Rename-BrokerCatalog cmdlet changes the name of a catalog. A catalog cannot have the same name as another catalog.

The following special characters are not allowed in a catalog name: \ / ; : # . * ? = < > | [ ] ( ) “‘

Examples

EXAMPLE 1

Renames the catalog with the name “Old Name”to “New Name”.

```
Rename-BrokerCatalog -Name "Old Name" -NewName "New Name"
```

EXAMPLE 2

Renames the catalog with the name “Old Name”to “New Name”.

```
c:\$catalog = Get-BrokerCatalog -Name "Old Name"
Rename-BrokerCatalog -InputObject $catalog -NewName "New Name"
```

Parameters

-InputObject

Specifies the catalog to rename.

Type:	Catalog[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the catalog to rename.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the catalog.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

You can pipe catalogs to Rename-BrokerCatalog.

Outputs

None or Citrix.Broker.Admin.SDK.Catalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Catalog object.

Related Links

- Get-BrokerCatalog
- New-BrokerCatalog
- Remove-BrokerCatalog
- Set-BrokerCatalog
- Move-BrokerCatalog

Rename-BrokerCatalogRebootSchedule

March 11, 2024

Renames a catalog reboot schedule.

Syntax

```
Rename-BrokerCatalogRebootSchedule
      [-InputObject] <CatalogRebootSchedule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerCatalogRebootSchedule
      [-Name] <String>
      [-PassThru]
```



```
4         [-NewName] <String>
5         [-LoggingId <Guid>]
6         [<CitrixCommonParameters>]
7         [<CommonParameters>]
```

Description

The Rename-BrokerCatalogRebootSchedule cmdlet changes the name of a catalog reboot schedule. A reboot schedule cannot have the same name as another reboot schedule.

Examples

EXAMPLE 1

Renames the catalog reboot schedule with name “Old Name”to “New Name”.

```
1  Rename-BrokerCatalogRebootSchedule -Name "Old Name" -NewName "New Name"
```

EXAMPLE 2

Renames catalog reboot schedule with the Uid 1 to “New Name”, showing the result.

```
1  Get-BrokerCatalogRebootSchedule -Uid 1 | Rename-
       BrokerCatalogRebootSchedule -NewName "New Name" -PassThru
```

Parameters

-InputObject

Specifies the reboot schedule to rename.

Type:	CatalogRebootSchedule[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the reboot schedule to rename.

Type:	String
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name for the reboot schedule.

Type:	String
Position:	3
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.CatalogRebootSchedule

You can pipe reboot schedules into Rename-BrokerCatalogRebootSchedule.

Outputs

None or Citrix.Broker.Admin.SDK.CatalogRebootSchedule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.CatalogRebootSchedule object.

Related Links

- Get-BrokerRebootScheduleV2
- Set-BrokerRebootScheduleV2
- New-BrokerRebootScheduleV2
- Remove-BrokerRebootScheduleV2
- Stop-BrokerRebootCycle

Rename-BrokerDesktopGroup

March 11, 2024

Renames a desktop group.

Syntax

```
Rename-BrokerDesktopGroup
       [-InputObject] <DesktopGroup[]>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-BrokerDesktopGroup
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-BrokerDesktopGroup cmdlet changes the name of a desktop group. A desktop group cannot have the same name as another desktop group.

Examples

EXAMPLE 1

Renames desktop group with the name “Old Name”to “New Name”.

```
Rename-BrokerDesktopGroup -Name "Old Name" -NewName "New Name"
```

EXAMPLE 2

Renames desktop group with the Uid 1 to “New Name”, showing the result.

```
Get-BrokerDesktopGroup -Uid 1 | Rename-BrokerDesktopGroup -NewName "New
    Name" -PassThru
```

Parameters

-InputObject

Specifies the desktop group to rename.

Type:	DesktopGroup[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the desktop group to rename.

Type:	String
Position:	2
Default value:	Null
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name that the desktop group will have.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

You can pipe desktop groups to Rename-BrokerDesktopGroup.

Outputs

None or Citrix.Broker.Admin.SDK.DesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.DesktopGroup object.

Notes

Renaming a desktop group does not alter its published name. If you need to change the name with which this desktop group appears to end-users, set a new value for the PublishedName property using the Set-BrokerDesktopGroup cmdlet.

Related Links

- about_Broker_Desktops
- Get-BrokerDesktopGroup
- New-BrokerDesktopGroup
- Set-BrokerDesktopGroup
- Move-BrokerDesktopGroup
- Remove-BrokerDesktopGroup

Rename-BrokerEntitlementPolicyRule

March 11, 2024

Renames a desktop rule in the site's entitlement policy.

Syntax

```
Rename-BrokerEntitlementPolicyRule
      [-InputObject] <EntitlementPolicyRule[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerEntitlementPolicyRule
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Rename-BrokerEntitlementPolicyRule cmdlet renames a desktop rule in the site's entitlement policy. The Name property of the rule is changed.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

Examples

EXAMPLE 1

Renames the desktop rule in the entitlement policy called Prod Dev to Product Development. The new name of the rule must be unique in the entitlement policy.

```
Rename-BrokerEntitlementPolicyRule 'Prod Dev' -NewName 'Product
    Development'
```

Parameters

-InputObject

The desktop rule in the entitlement policy to be renamed.

Type:	EntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The existing name of the desktop rule in the entitlement policy to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name of the desktop rule in the entitlement policy being renamed. The new name must not match that of any other existing rule in the policy (irrespective of whether it is a desktop or application rule).

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

The desktop rule in the entitlement policy being renamed.

Outputs

None or Citrix.Broker.Admin.SDK.EntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.EntitlementPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_ErrorHandling
- New-BrokerEntitlementPolicyRule

- Get-BrokerEntitlementPolicyRule
- Set-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule

Rename-BrokerGpoPolicy

March 11, 2024

Rename policies of the same name.

Syntax

```
Rename-BrokerGpoPolicy
      [-InputObject] <GpoPolicy[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerGpoPolicy
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Renames GPO policies with the supplied name. Policy names are unique in a policy set. Policies of the same name can exist in different policy sets. This cmdlet renames all policies in different policy sets.

Examples

EXAMPLE 1

Renames all the policies with the name ‘Test’to ‘Policy1’.

```
Rename-BrokerGpoPolicy -Name 'Test' -NewName 'Policy1'
```


Parameters

-InputObject

Specifies the GPO policy objects to rename.

Type:	GpoPolicy[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Identifies the policies to be renamed by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies new name for the policies.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoPolicy

The policy to rename can be piped into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.GpoPolicy

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.GpoPolicy object.

Notes

Note that when renaming a policy, its Guid remains the same and any associations are maintained.

Related Links

- Get-BrokerGpoPolicy
- New-BrokerGpoPolicy
- Remove-BrokerGpoPolicy
- Set-BrokerGpoPolicy

Rename-BrokerGpoPolicySet

March 11, 2024

Rename a GPO policy set.

Syntax

```
Rename-BrokerGpoPolicySet
      [-InputObject] <GpoPolicySet[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerGpoPolicySet
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Renames a GPO policy set to a new name.

Examples

EXAMPLE 1

Renames the policy set with the name ‘Test’to ‘Set1’.

```
Rename-BrokerGpoPolicySet -Name 'Test' -NewName 'Set1'
```

Parameters

-InputObject

Specifies the GPO policy set object to rename.

Type:	GpoPolicySet[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

Identifies the policy set to be renamed by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies new name for the policy set.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoPolicySet

The policy set to rename can be piped into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.GpoPolicySet

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.GpoPolicySet object.

Notes

Note that when renaming a policy set, its Guid remains the same and any associations are maintained.

Related Links

- Get-BrokerGpoPolicySet
- New-BrokerGpoPolicySet
- Remove-BrokerGpoPolicySet
- Set-BrokerGpoPolicySet

Rename-BrokerImportDb

March 11, 2024

This cmdlet is for internal use only

Syntax

```
Rename-BrokerImportDb
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

String

Related Links

Rename-BrokerMachineConfiguration

March 11, 2024

Renames a machine configuration.

Syntax

```
Rename-BrokerMachineConfiguration
      [-InputObject] <MachineConfiguration[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Rename-BrokerMachineConfiguration
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-MachineConfiguration cmdlet changes the name of a machine configuration. A machine configuration cannot have the same name as another machine configuration associated with the same slot.

Examples

EXAMPLE 1

Renames the machine configuration named “UPM\All Departments”to “UPM\Finance Department”

.

```
Rename-BrokerMachineConfiguration -Name "UPM\All Departments" -NewName
    "UPM\Finance Department"
```

EXAMPLE 2

Renames the machine configuration named “UPM\All Departments”to “UPM\Finance Department”

.

```
Rename-BrokerMachineConfiguration -Name "UPM\All Departments" -NewName
    "Finance Department"
```

Parameters

-InputObject

Machine configuration to rename.

Type:	MachineConfiguration[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Current name of machine configuration.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

New name for machine configuration. This may have the form "ConfigurationSlotName\MachineConfigurationNam
or "MachineConfigurationName". If the "ConfigurationSlotName"is provided it must match the name
of the configuration slot that the machine configuration is associated with.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.MachineConfiguration

Machine configuration to rename.

Outputs

None or Citrix.Broker.Admin.SDK.MachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.MachineConfiguration object.

Notes

The configuration slot can not be changed. Thus the left term of the Name and NewName must match.

Related Links

- New-BrokerMachineConfiguration
- Get-BrokerMachineConfiguration
- Set-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

Rename-BrokerPowerTimeScheme

March 11, 2024

Changes the name of an existing power time scheme.

Syntax


```
Rename-BrokerPowerTimeScheme
      [-InputObject] <PowerTimeScheme[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-BrokerPowerTimeScheme
      [-Name] <String>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Rename-BrokerPowerTimeScheme cmdlet renames a particular power time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times. In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see 'help about_Broker_PowerManagement'.

Examples

EXAMPLE 1

Renames the power time scheme named 'Development Weekdays'to 'Dev Week'.

```
Rename-BrokerPowerTimeScheme -Name 'Development Weekdays' -NewName 'Dev
    Week'
```

Parameters

-InputObject

The power time scheme to be renamed.

Type:	PowerTimeScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The current name of the power time scheme to be renamed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

The new name to be applied to the power time scheme.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.PowerTimeScheme

The power time scheme to be renamed.

Outputs

None or Citrix.Broker.Admin.SDK.PowerTimeScheme

This cmdlet does not generate any output, unless

you use the PassThru parameter, in which case it generates a

Citrix.Broker.Admin.SDK.PowerTimeScheme object.

Related Links

- about_Broker_PowerManagement
- Get-BrokerPowerTimeScheme
- Set-BrokerPowerTimeScheme
- New-BrokerPowerTimeScheme
- Remove-BrokerPowerTimeScheme

Rename-BrokerRebootScheduleV2

March 11, 2024

Renames a reboot schedule.

Syntax

```
Rename-BrokerRebootScheduleV2
      [-InputObject] <RebootScheduleV2[]>
      [-PassThru]
      [-NewName] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Rename-BrokerRebootScheduleV2
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-BrokerRebootScheduleV2 cmdlet changes the name of a reboot schedule. A reboot schedule cannot have the same name as another reboot schedule.

Examples

EXAMPLE 1

Renames the reboot schedule with name “Old Name”to “New Name”.

```
Rename-BrokerRebootScheduleV2 -Name "Old Name" -NewName "New Name"
```

EXAMPLE 2

Renames reboot schedule with the Uid 1 to “New Name”, showing the result.

```
Get-BrokerRebootScheduleV2 -Uid 1 | Rename-BrokerRebootScheduleV2 -
    NewName "New Name" -PassThru
```

Parameters

-InputObject

Specifies the reboot schedule to rename.

Type:	RebootScheduleV2[]
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

Specifies the name of the reboot schedule to rename.

Type:	String
Position:	2
Default value:	Null
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name for the reboot schedule.

Type:	String
Position:	3
Default value:	Null
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootScheduleV2

You can pipe reboot schedules into Rename-BrokerRebootScheduleV2.

Outputs

None or Citrix.Broker.Admin.SDK.RebootScheduleV2

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.RebootScheduleV2 object.

Related Links

- Get-BrokerRebootScheduleV2
- Set-BrokerRebootScheduleV2
- New-BrokerRebootScheduleV2
- Remove-BrokerRebootScheduleV2
- Stop-BrokerRebootCycle

Rename-BrokerTag

March 11, 2024

Rename one or more tags.

Syntax

```
Rename-BrokerTag
       [-InputObject] <Tag[]>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-BrokerTag
       [-Name] <String>
       [-PassThru]
       [-NewName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Renames one or more tags with the supplied name.

Examples

EXAMPLE 1

Renames tags with the name 'OldName'to 'ReplacementName'.

```
Rename-BrokerTag -Name 'OldName' -NewName 'ReplacementName'
```

Parameters

-InputObject

Specifies one or more tag objects to rename.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Identifies tags to be renamed by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies new name for the tags.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Tag

The tag to rename can be piped into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.Tag

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Tag object.

Notes

Note that when renaming a tag, its UUID remains the same and any associations are maintained.

Related Links

- Add-BrokerTag
- Get-BrokerTag
- New-BrokerTag

- Remove-BrokerTag
- Set-BrokerTag

Reset-BrokerEnabledFeatureList

March 11, 2024

Refreshes the Broker service's list of enabled features.

Syntax

```
Reset-BrokerEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Broker Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Broker service instance's list of enabled features.

```
Reset-BrokerEnabledFeatureList
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-BrokerHypervisorConnection

March 11, 2024

Reset the hypervisor connection

Syntax

```
Reset-BrokerHypervisorConnection
      [[-HypervisorConnectionUid] <Int32>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Reset-BrokerHypervisorConnection
     [-InputObject] <HypervisorConnection>
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Requests the hypervisor connection to be reset. The connection is dropped, details including credentials refreshed and the connection re-established. The reset request is asynchronous and may take a moment to occur

Examples

EXAMPLE 1

This command resets the specified Hypervisor connection by internal Uid

```
Reset-BrokerHypervisorConnection -HypervisorConnectionUid 2
```

EXAMPLE 2

This command resets the piped in hypervisor connection

```
Get-HypervisorConnection | Reset-BrokerHypervisorConnection
```

Parameters

-InputObject

Specifies the hypervisor connection object to remove.

Type:	HypervisorConnection
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-HypervisorConnectionUid

Specifies the hypervisor connection Uid

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.HypervisorConnection

You can pipe the hypervisor connection to be reset to Reset-BrokerHypervisorConnection.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerHypervisorConnection
- Remove-BrokerHypervisorConnection
- Set-BrokerHypervisorConnection

Reset-BrokerLhcDbInstance

March 11, 2024

This cmdlet is for internal use only

Syntax

```
Reset-BrokerLhcDbInstance
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet accepts no input

Outputs

String

Related Links

Reset-BrokerLicensingConnection

March 11, 2024

Resets the broker's license server connection.

Syntax

```
Reset-BrokerLicensingConnection
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Reset-BrokerLicensingConnection cmdlet resets the broker's connection to the license server.

Licensing changes resulting from new license files or alterations to the site-level licensing properties don't become effective immediately. There will typically be a delay as the changes are propagated across the site based on the scheduling of refresh logic built into the controllers and the license server.

Resetting the connection causes the list of available licenses for the connection to be updated. After adding licenses or changing the site-level licensing properties you can run Reset-BrokerLicensingConnection to ensure that the broker can access the new licenses immediately.

Each broker service instance holds its own connection to the license server. In order for the licensing changes to be applied immediately throughout the XenDesktop site this command needs to be run on every controller in the site.

Examples

EXAMPLE 1

Reset the broker's license server connection.

```
Reset-BrokerLicensingConnection
```

Parameters

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerSite
- Set-BrokerSite

Reset-BrokerServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Broker Service.

Syntax

```
Reset-BrokerServiceGroupMembership
      [-ConfigServiceInstance] <PSObject[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Broker Service access permissions and configuration service locations. The Reset-BrokerServiceGroupMembership command must be run on at least one instance of the service type (Broker) after installation and registration with the configuration service. Without this operation, the Broker services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional

services are added to the deployment, provided that the configuration service is not stopped. The Reset-BrokerServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    BrokerServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-BrokerServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	PSObject[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-BrokerServiceGroupMembership command.

Outputs

Citrix.Broker.Sdk.ServiceInstance

Reset-BrokerServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Broker Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerServiceInstance
- Get-BrokerServiceStatus

Send-BrokerSessionMessage

March 11, 2024

Sends a message to a session.

Syntax

```
Send-BrokerSessionMessage
    [-InputObject] <Session[]>
    [-MessageStyle] <SendMessageStyle>
    [-Title] <String>
    [-Text] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Generates a message box in the target session(s).

Examples

EXAMPLE 1

Sends a message to all the sessions for any user in MYDOMAIN.

```
$sessions = Get-BrokerSession -UserName MYDOMAIN\*
Send-BrokerSessionMessage $sessions -MessageStyle Information -Title
   TestTitle -Text TestMessage
```

EXAMPLE 2

Sends a message to the session on the desktop with Uid 1.

```
$desktop = Get-BrokerDesktop -Uid 1
Send-BrokerSessionMessage $desktop.SessionUid -MessageStyle Information
    -Title TestTitle -Text TestMessage
```


EXAMPLE 3

Trap and display error information.

```
trap  [Citrix.Broker.Admin.SDK.SdkOperationException]
{

  write $("Exception name = " + $_.Exception.GetType().FullName)
  write $("SdkOperationException.Status = " + $_.Exception.Status)
  write $("SdkOperationException.ErrorData=")
  $_.Exception.ErrorData

  write $("SdkOperationException.InnerException = " + $_.Exception.
     InnerException)
  $_.Exception.InnerException
  continue
 }


Send-BrokerSessionMessage -InputObject 10,11,12 -MessageStyle
   Information -Title "message title" -Text "message text"
```

Parameters

-InputObject

The target session(s) to send the message to.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MessageStyle

The style of message box to use (valid values are Critical, Question, Exclamation, or Information).

Type:	SendMessageStyle
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Title

Text to display in the messagebox title bar.

Type:	String
Position:	4
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Text

The message to display.

Type:	String
Position:	5
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The session to which to send the message can be piped in.

Outputs

None

By default, this cmdlet returns no output.

Notes

Sessions can be passed as the InputObject parameter as either session objects or their numeric Uids.

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between controller and machine, if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Send-BrokerSessionMessage could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the send message operation is invoked on the machines running valid sessions.

The system can fail to invoke the operation if the machine is not in an appropriate state or if there are problems in communicating with the machine. When an operation is invoked the system detects if the operation was initiated successfully or not by the session. As this operation is non-blocking the system doesn't detect or report whether the operation ultimately succeeded or failed after its successful initialization in the session.

Operation failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandli
In the event of errors the SdkErrorRecord error status code is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.
- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked in the session.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed in the session.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:

"Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:<OperationsSucceededCount>"

Related Links

- Get-BrokerSession

Set-BrokerAccessPolicyRule

March 11, 2024

Modifies an existing rule in the site's access policy.

Syntax

```
Set-BrokerAccessPolicyRule
    [-InputObject] <AccessPolicyRule[]>
    [-PassThru]
    [-AddExcludedClientIPs <IPAddressRange[]>]
    [-AddExcludedClientNames <String[]>]
    [-AddExcludedSmartAccessTags <String[]>]
    [-AddExcludedUsers <User[]>]
    [-AddIncludedClientIPs <IPAddressRange[]>]
    [-AddIncludedClientNames <String[]>]
    [-AddIncludedSmartAccessTags <String[]>]
    [-AddIncludedUsers <User[]>]
    [-AllowedConnections <AllowedConnection>]
    [-AllowedProtocols <String[]>]
    [-AllowedUsers <AllowedUser>]
    [-AllowRestart <Boolean>]
    [-AppProtectionKeyLoggingRequired <Boolean>]
    [-AppProtectionScreenCaptureRequired <Boolean>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-ExcludedClientIPFilterEnabled <Boolean>]
    [-ExcludedClientIPs <IPAddressRange[]>]
    [-ExcludedClientNameFilterEnabled <Boolean>]
    [-ExcludedClientNames <String[]>]
    [-ExcludedSmartAccessFilterEnabled <Boolean>]
    [-ExcludedSmartAccessTags <String[]>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-HdxSslEnabled <Boolean>]
    [-IncludedClientIPFilterEnabled <Boolean>]
    [-IncludedClientIPs <IPAddressRange[]>]
    [-IncludedClientNameFilterEnabled <Boolean>]
    [-IncludedClientNames <String[]>]
    [-IncludedSmartAccessFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterType <String>]
    [-IncludedSmartAccessTags <String[]>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-RemoveExcludedClientIPs <IPAddressRange[]>]
    [-RemoveExcludedClientNames <String[]>]
    [-RemoveExcludedSmartAccessTags <String[]>]
    [-RemoveExcludedUsers <User[]>]
    [-RemoveIncludedClientIPs <IPAddressRange[]>]
    [-RemoveIncludedClientNames <String[]>]
    [-RemoveIncludedSmartAccessTags <String[]>]
    [-RemoveIncludedUsers <User[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```



```
Set-BrokerAccessPolicyRule
    [-Name] <String>
    [-PassThru]
    [-AddExcludedClientIPs <IPAddressRange[]>]
    [-AddExcludedClientNames <String[]>]
    [-AddExcludedSmartAccessTags <String[]>]
    [-AddExcludedUsers <User[]>]
    [-AddIncludedClientIPs <IPAddressRange[]>]
    [-AddIncludedClientNames <String[]>]
    [-AddIncludedSmartAccessTags <String[]>]
    [-AddIncludedUsers <User[]>]
    [-AllowedConnections <AllowedConnection>]
    [-AllowedProtocols <String[]>]
    [-AllowedUsers <AllowedUser>]
    [-AllowRestart <Boolean>]
    [-AppProtectionKeyLoggingRequired <Boolean>]
    [-AppProtectionScreenCaptureRequired <Boolean>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-ExcludedClientIPFilterEnabled <Boolean>]
    [-ExcludedClientIPs <IPAddressRange[]>]
    [-ExcludedClientNameFilterEnabled <Boolean>]
    [-ExcludedClientNames <String[]>]
    [-ExcludedSmartAccessFilterEnabled <Boolean>]
    [-ExcludedSmartAccessTags <String[]>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-HdxSslEnabled <Boolean>]
    [-IncludedClientIPFilterEnabled <Boolean>]
    [-IncludedClientIPs <IPAddressRange[]>]
    [-IncludedClientNameFilterEnabled <Boolean>]
    [-IncludedClientNames <String[]>]
    [-IncludedSmartAccessFilterEnabled <Boolean>]
    [-IncludedSmartAccessFilterType <String>]
    [-IncludedSmartAccessTags <String[]>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-RemoveExcludedClientIPs <IPAddressRange[]>]
    [-RemoveExcludedClientNames <String[]>]
    [-RemoveExcludedSmartAccessTags <String[]>]
    [-RemoveExcludedUsers <User[]>]
    [-RemoveIncludedClientIPs <IPAddressRange[]>]
    [-RemoveIncludedClientNames <String[]>]
    [-RemoveIncludedSmartAccessTags <String[]>]
    [-RemoveIncludedUsers <User[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

The Set-BrokerAccessPolicyRule cmdlet modifies an existing rule in the site's access policy.

An access policy rule defines a set of connection filters and access control rights relating to a desktop group. These allow fine-grained control of what access is granted to a desktop group based on details of, for example, a user's endpoint device, its address, and the user's identity.

Changing a rule does not affect existing user sessions, but it may result in users being unable to launch new sessions, or reconnect to disconnected sessions if the change removes access to the desktop group delivering those sessions.

Examples

EXAMPLE 1

Adds user group OFFICE\contractors to the Temp Staff access policy rule. The resources that the group can access are dependent on the existing properties of the rule in addition to the site's assignment and entitlement policies.

```
Set-BrokerAccessPolicyRule 'Temp Staff' -AddIncludedUsers office\
    contractors
```

EXAMPLE 2

Modifies the Temp Staff access policy rule to remove access to any user device with an IP address matching 10.15.0.0/16, and requires that all connections by the rule must come through Access Gateway (assuming that the included SmartAccess tags filter is enabled).

```
Set-BrokerAccessPolicyRule 'Temp Staff' -ExcludedClientIPFilterEnabled
    $true -AddExcludedClientIPs '10.15.0.0/16' -AllowedConnections ViaAG
```

Parameters

-InputObject

The access policy rule to be modified.

Type:	AccessPolicyRule[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the access policy rule to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedClientIPs

Adds the specified user device IP addresses to the excluded client IP address filter of the rule.

See the ExcludedClientIPs parameter for more information.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedClientNames

Adds the specified user device names to the excluded client names filter of the rule.

See the ExcludedClientNames parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedSmartAccessTags

Adds the specified SmartAccess tags to the excluded SmartAccess tags filter of the rule.

See the ExcludedSmartAccessTags parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedUsers

Adds the specified users and groups to the excluded users filter of the rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedClientIPs

Adds the specified user device IP addresses to the included client IP address filter of the rule.

See the IncludedClientIPs parameter for more information.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedClientNames

Adds the specified user device names to the included client names filter of the rule.

See the IncludedClientNames parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedSmartAccessTags

Adds the specified SmartAccess tags to the included SmartAccess tags filter of the rule.

See the IncludedSmartAccessTags parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedUsers

Adds the specified users and groups to the included users filter of the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowedConnections

Changes whether connections must be local or via Access Gateway, and if so whether specified SmartAccess tags must be provided by Access Gateway with the connection. This property forms part of the included SmartAccess tags filter.

Valid values are Filtered, NotViaAG, ViaAG and AnyViaAG.

For a detailed description of this property see "help about_Broker_AccessPolicy".

Type:	AllowedConnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowedProtocols

Changes the protocols (for example HDX, RDP) available to the user for sessions delivered from the rule's desktop group. If the user gains access to a desktop group by multiple rules, the allowed protocol list is the combination of the protocol lists from all those rules.

If the protocol list is empty, access to the desktop group is implicitly denied.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowedUsers

Changes the behavior of the included users filter of the rule. This can restrict access to a list of named users or groups, allow access to any authenticated user, any user (whether authenticated or not), or only non-authenticated users. For a detailed description of this property see "help about_Broker_AccessPolicy".

Valid values are Filtered, AnyAuthenticated, Any, AnonymousOnly and FilteredOrAnonymous.

Type:	AllowedUser

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowRestart

Changes whether the user can restart sessions delivered from the rule's desktop group. Session restart is handled as follows: For sessions on single-session power-managed machines, the machine is powered off, and a new session launch request made; for sessions on multi-session machines, a logoff request is issued to the session, and a new session launch request made; otherwise the property is ignored.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionKeyLoggingRequired

Specifies whether key logging app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionScreenCaptureRequired

Specifies whether screen capture app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Changes whether the rule is enabled or disabled. A disabled rule is ignored when evaluating the site's access policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ExcludedClientIPFilterEnabled

Changes whether the excluded client IP address filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedClientIPs

Changes the IP addresses of user devices explicitly denied access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the excluded client IP address filter.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedClientNameFilterEnabled

Changes whether the excluded client names filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedClientNames

Changes which names of user devices are explicitly denied access to the rule's desktop group. This property forms part of the excluded client names filter.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedSmartAccessFilterEnabled

Changes whether the excluded SmartAccess tags filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedSmartAccessTags

Changes which SmartAccess tags explicitly deny access to the rule's desktop group if any occur in those provided with the user's connection. This property forms part of the excluded SmartAccess tags filter.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Changes whether the excluded users filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUsers

Changes which users and groups are explicitly denied access to the rule's desktop group. This property forms part of the excluded users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-HdxSslEnabled

Indicates whether TLS encryption is enabled for sessions delivered from the rule's desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientIPFilterEnabled

Changes whether the included client IP address filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientIPs

Changes which IP addresses of user devices allowed access to the rule's desktop group. Addresses can be specified as simple numeric addresses or as subnet masks (for example, 10.40.37.5 or 10.40.0.0/16). This property forms part of the included client IP address filter.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientNameFilterEnabled

Changes whether the included client name filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedClientNames

Changes which names of user devices are allowed access to the rule's desktop group. This property forms part of the included client names filter.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessFilterEnabled

Changes whether the included SmartAccess tags filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessFilterType

Changes whether all tags present in IncludedSmartAccessTags must match tags provided by the user's connection to grant access (MatchAll), or whether any tag matching is sufficient (MatchAny).

Type:	String
Accepted values:	MatchAll, MatchAny
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedSmartAccessTags

Changes which SmartAccess tags grant access to the rule's desktop group if they occur in those provided with the user's connection. If multiple tags are specified, access also depends on the IncludedSmartAccessFilterType setting. This property forms part of the included SmartAccess tags filter.

Type:	String[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Changes whether the included users filter is enabled or disabled. If the filter is disabled, it is ignored when the access policy rule is evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUsers

Changes which users and groups are granted access to the rule's desktop group. This property forms part of the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedClientIPs

Removes the specified user device IP addresses from the excluded client IP address filter of the rule.

See the ExcludedClientIPs parameter for more information.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedClientNames

Removes the specified user device names from the excluded client names filter of the rule.

See the ExcludedClientNames parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedSmartAccessTags

Removes the specified SmartAccess tags from the excluded SmartAccess tags filter of the rule.

See the ExcludedSmartAccessTags parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedUsers

Removes the specified users and groups from the excluded users filter of the rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedClientIPs

Removes the specified user device IP addresses from the included client IP address filter of the rule.

See the IncludedClientIPs parameter for more information.

Type:	IPAddressRange[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedClientNames

Removes the specified client names from the included client names filter of the rule.

See the IncludedClientNames parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedSmartAccessTags

Removes the specified SmartAccess tags from the included SmartAccess tags filter of the rule.

See the IncludedSmartAccessTags parameter for more information.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedUsers

Removes the specified users and groups from the included users filter of the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AccessPolicyRule

The access policy rule to be modified.

Outputs

None, or Citrix.Broker.Admin.SDK.AccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AccessPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_ErrorHandling
- New-BrokerAccessPolicyRule
- Get-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule
- Remove-BrokerAccessPolicyRule

Set-BrokerAccessPolicyRuleMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for AccessPolicyRule

Syntax

```
Set-BrokerAccessPolicyRuleMetadata
   [-AccessPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAccessPolicyRuleMetadata
   [-AccessPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAccessPolicyRuleMetadata
   [-AccessPolicyRuleId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerAccessPolicyRuleMetadata
   [-InputObject] <AccessPolicyRule[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAccessPolicyRuleMetadata
   [-InputObject] <AccessPolicyRule[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAccessPolicyRuleMetadata
   [-AccessPolicyRuleName] <String>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAccessPolicyRuleMetadata
   [-AccessPolicyRuleName] <String>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerAccessPolicyRuleMetadata cmdlet creates/updates metadata key-value pairs for AccessPolicyRule. The AccessPolicyRule can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the AccessPolicyRule whose instance is pointed by $obj-Uid


```
Set-BrokerAccessPolicyRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the AccessPolicyRule in the site

```
Get-BrokerAccessPolicyRule | Set-BrokerAccessPolicyRuleMetadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the AccessPolicyRule in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerAccessPolicyRuleMetadata 'objname'
```

Parameters

-AccessPolicyRuleId

Specifies the AccessPolicyRule object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the AccessPolicyRule objects whose Metadata is to be created/updated.

Type:	AccessPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-AccessPolicyRuleName

Specifies the AccessPolicyRule object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAccessPolicyRule

You can pipe the AccessPolicyRule to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerAccessPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerAccessPolicyRule object.

Related Links

Set-BrokerAdminFolderMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for AdminFolder

Syntax

```
Set-BrokerAdminFolderMetadata
   [-AdminFolderId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-AdminFolderId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-AdminFolderId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-InputObject] <AdminFolder[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-InputObject] <AdminFolder[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-AdminFolderName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAdminFolderMetadata
    [-AdminFolderName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerAdminFolderMetadata cmdlet creates/updates metadata key-value pairs for AdminFolder. The AdminFolder can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the AdminFolder whose instance is pointed by $obj-Uid

```
Set-BrokerAdminFolderMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the AdminFolder in the site

```
Get-BrokerAdminFolder | Set-BrokerAdminFolderMetadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the AdminFolder in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerAdminFolderMetadata 'objname'
```

Parameters

-AdminFolderId

Specifies the AdminFolder object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the AdminFolder objects whose Metadata is to be created/updated.

Type:	AdminFolder[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-AdminFolderName

Specifies the AdminFolder object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAdminFolder

You can pipe the AdminFolder to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerAdminFolder

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerAdminFolder object.

Related Links

Set-BrokerAppAssignmentPolicyRule

March 11, 2024

Modifies an existing application rule in the site's assignment policy.

Syntax

```
Set-BrokerAppAssignmentPolicyRule
   [-InputObject] <AppAssignmentPolicyRule[]>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-RemoveExcludedUsers <User[]>]
   [-RemoveIncludedUsers <User[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAppAssignmentPolicyRule
   [-Name] <String>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-RemoveExcludedUsers <User[]>]
   [-RemoveIncludedUsers <User[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

The Set-BrokerAppAssignmentPolicyRule cmdlet modifies an existing application rule in the site's assignment policy.

An application rule in the assignment policy defines the users who are entitled to a self-service persistent machine assignment from the rule's desktop group; once assigned the machine can run one or more applications published from the group.

Changing an application rule does not alter machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Examples

EXAMPLE 1

Adds the user group OFFICE\interns to the Temp Staff application rule in the assignment policy. This grants all members of that user group an entitlement to a machine in the rule's desktop group. The machines can run applications published from the group. The application session properties obtained using the rule are determined by the rule's other properties.

```
Set-BrokerAppAssignmentPolicyRule 'Temp Staff' -AddIncludedUsers office
    \interns
```

EXAMPLE 2

Disables the Temp Staff application rule in the assignment policy. This prevents further machine assignments being made using this rule until it is re-enabled. However, access to machines already assigned using the rule is not impacted.

```
Set-BrokerAppAssignmentPolicyRule 'Temp Staff' -Enabled $false
```

Parameters

-InputObject

The application rule in the assignment policy to be modified.

Type:	AppAssignmentPolicyRule[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application rule in the assignment policy to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedUsers

Adds the specified users to the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedUsers

Adds the specified users to the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the application rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Enables or disables the application rule. A disabled rule is ignored when evaluating the site's assignment policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUsers

Changes the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from the rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUsers

Changes the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule, or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedUsers

Removes the specified users from the excluded users filter of the application rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedUsers

Removes the specified users from the included users filter of the application rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

The application rule within the assignment policy to be modified.

Outputs

None or Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AppAssignmentPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppAssignmentPolicyRule
- Get-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule
- Remove-BrokerAppAssignmentPolicyRule

Set-BrokerAppEntitlementPolicyRule

March 11, 2024

Modifies an existing application rule in the site's entitlement policy.

Syntax

```
Set-BrokerAppEntitlementPolicyRule
   [-InputObject] <AppEntitlementPolicyRule[]>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-LeasingBehavior <LeasingBehavior>]
   [-RemoveExcludedUsers <User[]>]
   [-RemoveIncludedUsers <User[]>]
   [-SessionReconnection <SessionReconnection>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAppEntitlementPolicyRule
   [-Name] <String>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
```



```
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-LeasingBehavior <LeasingBehavior>]
    [-RemoveExcludedUsers <User[]>]
    [-RemoveIncludedUsers <User[]>]
    [-SessionReconnection <SessionReconnection>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerAppEntitlementPolicyRule cmdlet modifies an existing application rule in the site's entitlement policy.

An application rule in the entitlement policy defines the users who are allowed per-session access to a machine to run one or more applications published from the rule's desktop group.

Changing a rule does not affect existing sessions launched using the rule, but if the change removes an entitlement to a machine that was previously granted, users may be unable to reconnect to a disconnected session on that machine.

Examples

EXAMPLE 1

Adds the user group OFFICE\contractors to those entitled to run applications from the rule's associated desktop group. This grants all members of that group an entitlement to an application session from that group.

```
Set-BrokerAppEntitlementPolicyRule 'Temp Workers' -AddIncludedUsers
    office\contractors
```

EXAMPLE 2

Disables the Temp Workers application rule in the entitlement policy. This prevents further application sessions being launched using this rule until it is re-enabled. However, access to existing application sessions is not affected.

```
Set-BrokerAppEntitlementPolicyRule 'Temp Workers' -Enabled $false
```


Parameters

-InputObject

The application rule in the entitlement policy to be modified.

Type:	AppEntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the application rule in the entitlement policy to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AddExcludedUsers

Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedUsers

Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the application rule. The text is purely informational for the administrator, it is never visible to the end user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Enables or disables the application rule. A disabled rule is ignored when evaluating the site's entitlement policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUsers

Changes the excluded users filter of the rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group.

This can be used to exclude users or groups or users who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to an application session by the application rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUsers

Changes the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeasingBehavior

Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:

Allowed and Disallowed.

The Allowed value indicates that connection leasing should behave normally. The Disallowed value prevents users

from launching or reconnecting to sessions using this entitlement while connection leasing is active (typically during a database outage).

Type:	LeasingBehavior
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedUsers

Removes the specified users from the excluded users filter of the application rule, that is, the users and groups who are explicitly denied entitlements to run applications published from the desktop group.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedUsers

Removes the specified users from the included users filter of the rule, that is, the users and groups who are granted an entitlement to an application session by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

The application rule in the entitlement policy rule to be modified.

Outputs

None or Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AppEntitlementPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_ErrorHandling
- New-BrokerAppEntitlementPolicyRule
- Get-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule
- Remove-BrokerAppEntitlementPolicyRule

Set-BrokerApplication

March 11, 2024

Changes the settings of an application to the value specified in the command.

Syntax

```
Set-BrokerApplication
   [-InputObject] <Application[]>
   [-PassThru]
   [-BrowserName <String>]
   [-ClientFolder <String>]
   [-CommandLineArguments <String>]
   [-CommandLineExecutable <String>]
   [-CpuPriorityLevel <CpuPriorityLevel>]
   [-Description <String>]
   [-DoNotEnumerate <Boolean>]
   [-Enabled <Boolean>]
   [-HomeZoneOnly <Boolean>]
   [-HomeZoneUid <Guid>]
   [-IconFromClient <Boolean>]
   [-IconUid <Int32>]
   [-IgnoreUserHomeZone <Boolean>]
   [-LocalLaunchDisabled <Boolean>]
   [-MaxPerMachineInstances <Int32>]
   [-MaxPerUserInstances <Int32>]
   [-MaxTotalInstances <Int32>]
   [-PublishedName <String>]
   [-SecureCmdLineArgumentsEnabled <Boolean>]
   [-ShortcutAddedToDesktop <Boolean>]
   [-ShortcutAddedToStartMenu <Boolean>]
   [-StartMenuFolder <String>]
   [-UserFilterEnabled <Boolean>]
   [-Visible <Boolean>]
   [-WaitForPrinterCreation <Boolean>]
```



```
    [-WorkingDirectory <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplication
    [-Name] <String>
    [-PassThru]
    [-BrowserName <String>]
    [-ClientFolder <String>]
    [-CommandLineArguments <String>]
    [-CommandLineExecutable <String>]
    [-CpuPriorityLevel <CpuPriorityLevel>]
    [-Description <String>]
    [-DoNotEnumerate <Boolean>]
    [-Enabled <Boolean>]
    [-HomeZoneOnly <Boolean>]
    [-HomeZoneUid <Guid>]
    [-IconFromClient <Boolean>]
    [-IconUid <Int32>]
    [-IgnoreUserHomeZone <Boolean>]
    [-LocalLaunchDisabled <Boolean>]
    [-MaxPerMachineInstances <Int32>]
    [-MaxPerUserInstances <Int32>]
    [-MaxTotalInstances <Int32>]
    [-PublishedName <String>]
    [-SecureCmdLineArgumentsEnabled <Boolean>]
    [-ShortcutAddedToDesktop <Boolean>]
    [-ShortcutAddedToStartMenu <Boolean>]
    [-StartMenuFolder <String>]
    [-UserFilterEnabled <Boolean>]
    [-Visible <Boolean>]
    [-WaitForPrinterCreation <Boolean>]
    [-WorkingDirectory <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerApplication cmdlet changes the value of one or more properties of an application, such as its CpuPriorityLevel or its CommandLineArguments, to the value specified in the command.

This cmdlet lets you change only the settings of the Application object, and not the relationships to other objects. For instance, it does not let you change which users can access this application, or change which desktop groups this application is published to. To do this, you need to remove the existing association, and then add a new association. The following example shows how to change the desktop group that an application is associated with from $group1 to $group2:

Remove-BrokerApplication -DesktopGroup $group1

Add-BrokerApplication -DesktopGroup $group2

You can change properties of HostedOnDesktop, InstalledOnClient and PublishedContent applications but it is not possible to change the ApplicationType. Also, the Name cannot be changed using this cmdlet; to do this, use the Rename-BrokerApplication cmdlet.

Examples

EXAMPLE 1

Modifies the application that has a Name of "Notepad"so that its description reads Windows Notepad.

```
Set-BrokerApplication -Name "Notepad" -Description 'Windows Notepad'
```

EXAMPLE 2

First gets the application with a BrowserName of "Calculator", then modifies that application (by supplying the application object in the first position) so that it is disabled for users.

```
$app = Get-BrokerApplication -BrowserName "Calculator"
Set-BrokerApplication -InputObject $app -Enabled $false
```

Parameters

-InputObject

Specifies the application to modify. The Uid of the application can also be substituted for the object.

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the application to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BrowserName

Specifies the BrowserName of the application.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ClientFolder

Specifies the folder that the application belongs to as the user sees it. This is the application folder displayed in the Citrix Online Plug-in, in Web Services, and also in the user's start menu. Subdirectories can be specified with '\' character. The following special characters are not allowed: / * ? < > | ":. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CommandLineArguments

Specifies the command-line arguments to use when launching the executable. This setting is ignored for applications of type PublishedContent.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CommandLineExecutable

Specifies the name of the executable file to launch.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-CpuPriorityLevel

Specifies the CPU priority for the launched executable. Valid values are: Low, BelowNormal, Normal, AboveNormal, and High. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	CpuPriorityLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Specifies the description of the application.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DoNotEnumerate

Specifies whether or not this application is returned to the user during enumeration.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Specifies whether or not this application can be launched.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HomeZoneOnly

Specifies whether if the preferred zone for launching the application is its home zone but no machine is available from that zone then the launch fails.

This can only be set if the application has a home zone preference specified.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HomeZoneUid

Specifies any home zone preference used when launching this application.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconFromClient

Specifies if the app icon should be retrieved from the application on the client. This is reserved for possible future use, and all applications of type HostedOnDesktop cannot set or change this value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Specifies which icon to use for this application. This application is visible both to the administrator (in the consoles) and also to the user. If no icon is specified, then a generic built-in application icon is used.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreUserHomeZone

Specifies that when launching the application and the user has a home zone specified then the user's home zone preference should be ignored.

This can only be set if the application does not itself have a home zone preference specified.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalLaunchDisabled

When launching a published application from within a published desktop, do not launch the application in that desktop session.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerMachineInstances

Specifies the maximum allowed concurrently running instances of the application that an individual machine can have. A value of zero allows unlimited usage subject to any site-wide limit.

Reducing the limit below the currently running number of instances does not cause any of those to be stopped.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerUserInstances

Specifies the maximum allowed concurrently running instances of the application that an individual user can have. A value of zero allows unlimited usage subject to any site-wide limit.

Reducing the limit below the currently running number of instances does not cause any of those to be stopped.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTotalInstances

Specifies the maximum allowed total of concurrently running instances of the application in the site. A value of zero allows unlimited usage.

Reducing the limit below the currently running number of instances does not cause any of those to be stopped.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Specifies the name seen by end users who have access to this application.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureCmdLineArgumentsEnabled

Specifies whether the command-line arguments should be secured. This is reserved for possible future use, and all applications of type HostedOnDesktop can only have this value set to true.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShortcutAddedToDesktop

Specifies whether or not a shortcut to the application should be placed on the user device.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShortcutAddedToStartMenu

Specifies whether a shortcut to the application should be placed in the user's start menu on their user device.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartMenuFolder

Specifies the name of the start menu folder that holds the application shortcut (if any). This is valid only for the Citrix Online Plug-in. Subdirectories can be specified with ‘' character. The following special characters are not allowed: / * ? < > | “:.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Specifies whether the application's user filter is enabled or disabled. Where the user filter is enabled, the application is only visible to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Visible

Specifies whether or not this application is visible to users. Note that it's possible for an application to be disabled and still visible.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WaitForPrinterCreation

Specifies whether or not the session waits for the printers to be created before allowing the end-user to interact with the session. Note that this property cannot be set for applications of type InstalledOnClient.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WorkingDirectory

Specifies from which working directory the executable is launched from. This setting is ignored for applications of type PublishedContent.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Application, or depends on parameter

You can pipe the application to be added to Set-BrokerApplication. You can also pipe some of the other parameters by name.

Outputs

None or Citrix.Broker.Admin.SDK.Application

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Application object.

Related Links

- about_Broker_Applications
- New-BrokerApplication
- Add-BrokerApplication
- Get-BrokerApplication
- Remove-BrokerApplication
- Rename-BrokerApplication
- Move-BrokerApplication

Set-BrokerApplicationGroup

March 11, 2024

Changes properties of application groups.

Syntax

```
Set-BrokerApplicationGroup
    [-InputObject] <ApplicationGroup[]>
    [-PassThru]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-RestrictToTag <String>]
    [-SessionSharingEnabled <Boolean>]
    [-SingleAppPerSession <Boolean>]
    [-UserFilterEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationGroup
    [-Name] <String>
    [-PassThru]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-RestrictToTag <String>]
    [-SessionSharingEnabled <Boolean>]
    [-SingleAppPerSession <Boolean>]
    [-UserFilterEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerApplicationGroup cmdlet changes the properties of one or more application groups. The changed properties and the new values of those properties are specified as parameters to Set-BrokerApplicationGroup.

This cmdlet cannot change the relationships of application group objects with other objects. For instance, it cannot change which users can access applications in this application group, or change which desktop groups this application group is published to. To do this, you need to remove the existing association, and then add a new association.

The following example shows how to change the desktop group that application group $application-Group is associated with from $desktopGroup1 to $desktopGroup2:

Remove-BrokerApplicationGroup $applicationGroup -DesktopGroup $desktopGroup1

Add-RemoveApplicationGroup $applicationGroup -DesktopGroup $desktopGroup2

Application groups may not be renamed using this cmdlet. To rename an application group, use Rename-BrokerApplicationGroup.

Examples

EXAMPLE 1

Prevent new instances of applications in the 'Helpdesk Apps'application group from being launched.

```
Set-BrokerApplicationGroup 'Helpdesk Apps' -Enabled $false
```

EXAMPLE 2

Disable the user filter on every application group in the system. (Other access control mechanisms, such as access policy and user filters on individual applications, still apply.)

```
Get-BrokerApplicationGroup | Set-BrokerApplicationGroup -
    UserFilterEnabled $false
```

Parameters

-InputObject

Specifies the application group whose properties should be altered. Its Uid can also be substituted for the object reference.

Type:	ApplicationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Alters the properties of application groups whose name matches the supplied pattern.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

The new description for the application group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Whether the application group's applications should be launchable by end users.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

Specifies the new values of the application group's tag restriction.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionSharingEnabled

Specifies the new value of the application group's SessionSharingEnabled flag. Please note this setting and SingleAppPerSession cannot be true at the same time.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SingleAppPerSession

Specifies whether each application launched from this application group starts in its own new session or can share an existing suitable session if present. Please note this setting and SessionSharingEnabled cannot be true at the same time.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Specifies the new value of the application group's UserFilterEnabled flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.ApplicationGroup

You can pipe application groups to Set-BrokerApplication.

Outputs

None or Citrix.Broker.Admin.SDK.ApplicationGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.ApplicationGroup object.

Related Links

- about_Broker_Applications
- Add-BrokerApplicationGroup
- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Remove-BrokerApplicationGroup
- Rename-BrokerApplicationGroup
- Move-BrokerApplicationGroup

Set-BrokerApplicationGroupMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for ApplicationGroup

Syntax

```
Set-BrokerApplicationGroupMetadata
   [-ApplicationGroupId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
   [-ApplicationGroupId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
   [-ApplicationGroupId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
   [-InputObject] <ApplicationGroup[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
   [-InputObject] <ApplicationGroup[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
    [-ApplicationGroupName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationGroupMetadata
    [-ApplicationGroupName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerApplicationGroupMetadata cmdlet creates/updates metadata key-value pairs for ApplicationGroup. The ApplicationGroup can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the ApplicationGroup whose instance is pointed by $obj-Uid

```
Set-BrokerApplicationGroupMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the ApplicationGroup in the site

```
Get-BrokerApplicationGroup | Set-BrokerApplicationGroupMetadata -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the ApplicationGroup in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerApplicationGroupMetadata 'objname'
```

Parameters

-ApplicationGroupId

Specifies the ApplicationGroup object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the ApplicationGroup objects whose Metadata is to be created/updated.

Type:	ApplicationGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ApplicationGroupName

Specifies the ApplicationGroup object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplicationGroup

You can pipe the ApplicationGroup to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerApplicationGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerApplicationGroup object.

Related Links

Set-BrokerApplicationInstanceMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for ApplicationInstance

Syntax

```
Set-BrokerApplicationInstanceMetadata
   [-ApplicationInstanceId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationInstanceMetadata
   [-ApplicationInstanceId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerApplicationInstanceMetadata
   [-ApplicationInstanceId] <Int64>
```



```
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationInstanceMetadata
    [-InputObject] <ApplicationInstance[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationInstanceMetadata
    [-InputObject] <ApplicationInstance[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerApplicationInstanceMetadata cmdlet creates/updates metadata key-value pairs for ApplicationInstance. The ApplicationInstance can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the ApplicationInstance whose instance is pointed by $obj-Uid

```
Set-BrokerApplicationInstanceMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the ApplicationInstance in the site

```
Get-BrokerApplicationInstance | Set-BrokerApplicationInstanceMetadata -
    Name "MyMetadataName" -Value "1234"
```


Parameters

-ApplicationInstanceId

Specifies the ApplicationInstance object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the ApplicationInstance objects whose Metadata is to be created/updated.

Type:	ApplicationInstance[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplicationInstance

You can pipe the ApplicationInstance to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerApplicationInstance

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerApplicationInstance object.

Related Links

Set-BrokerApplicationMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Application

Syntax

```
Set-BrokerApplicationMetadata
    [-ApplicationId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-ApplicationId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-ApplicationId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-InputObject] <Application[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-InputObject] <Application[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-ApplicationName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerApplicationMetadata
    [-ApplicationName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerApplicationMetadata cmdlet creates/updates metadata key-value pairs for Application. The Application can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Application whose instance is pointed by $obj-Uid

```
Set-BrokerApplicationMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Application in the site

```
Get-BrokerApplication | Set-BrokerApplicationMetadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Application in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerApplicationMetadata 'objname'
```


Parameters

-ApplicationId

Specifies the Application object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Application objects whose Metadata is to be created/updated.

Type:	Application[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ApplicationName

Specifies the Application object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerApplication

You can pipe the Application to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerApplication

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerApplication object.

Related Links

Set-BrokerAssignmentPolicyRule

March 11, 2024

Modifies an existing desktop rule in the site's assignment policy.

Syntax

```
Set-BrokerAssignmentPolicyRule
   [-InputObject] <AssignmentPolicyRule[]>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IconUid <Int32>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-MaxDesktops <Int32>]
   [-PublishedName <String>]
   [-RemoveExcludedUsers <User[]>]
   [-RemoveIncludedUsers <User[]>]
   [-SecureIcaRequired <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRule
   [-Name] <String>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IconUid <Int32>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-MaxDesktops <Int32>]
   [-PublishedName <String>]
   [-RemoveExcludedUsers <User[]>]
```



```
    [-RemoveIncludedUsers <User[]>]
    [-SecureIcaRequired <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerAssignmentPolicyRule cmdlet modifies an existing desktop rule in the site's assignment policy.

A desktop rule in the assignment policy defines the users who are entitled to self-service persistent machine assignments from the rule's desktop group. A rule defines how many machines a user is allowed from the group for delivery of full desktop sessions.

Changing a desktop rule does not alter machine assignments that have already been made by the rule, nor does it affect active sessions to those machines in any way.

Examples

EXAMPLE 1

Adds the user group OFFICE\interns to the Temp Staff desktop rule in the assignment policy. This grants all members of that user group entitlements to machines in the rule's desktop group. The number of machine entitlements per user and the session properties of the desktops obtained using the rule are determined by the rule's other properties.

```
Set-BrokerAssignmentPolicyRule 'Temp Staff' -AddIncludedUsers office\
    interns
```

EXAMPLE 2

Disables the Temp Staff desktop rule in the assignment policy. This prevents further machine assignments being made using this rule until it is re-enabled. However, access to machines already assigned using the rule is not impacted.

```
Set-BrokerAssignmentPolicyRule 'Temp Staff' -Enabled $false
```


Parameters

-InputObject

The desktop rule in the assignment policy to be modified.

Type:	AssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the desktop rule in the assignment policy to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AddExcludedUsers

Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedUsers

Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Changes the color depth of any desktop sessions to machines assigned by the rule.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Enables or disables the desktop rule. A disabled rule is ignored when evaluating the site's assignment policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when assignment policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUsers

Changes the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Changes the unique ID of the icon used to display the machine assignment entitlement to the user, and of the assigned desktop itself following the assignment.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a machine assignment by the rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

For rules that relate to RemotePC desktop groups however, if the included user filter is disabled, the rule is effectively disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUsers

Changes the included users filter of the rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

If a user appears explicitly in the excluded users filter of the rule, or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxDesktops

The number of machines from the rule's desktop group to which a user is entitled. Where an entitlement is granted to a user group rather than an individual, the number of machines applies to each member of the user group independently.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Changes the published name of the machine assignment entitlement as seen by the user. Changing the published name does not affect the names of desktops that have already been assigned by the rule.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-RemoveExcludedUsers

Removes the specified users from the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a machine assignment from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedUsers

Removes the specified users from the included users filter of the desktop rule, that is, the users and groups who are granted an entitlement to a machine assignment by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Changes whether the desktop rule requires the SecureICA protocol for desktop sessions to machines assigned using the entitlement.

The default null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AssignmentPolicyRule

The desktop rule within the assignment policy to be modified.

Outputs

None or Citrix.Broker.Admin.SDK.AssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.AssignmentPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_AssignmentPolicy
- about_Broker_ErrorHandling
- New-BrokerAssignmentPolicyRule
- Get-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule
- Remove-BrokerAssignmentPolicyRule

Set-BrokerAssignmentPolicyRuleMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for AssignmentPolicyRule

Syntax

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-AssignmentPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerAssignmentPolicyRuleMetadata
   [-AssignmentPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-AssignmentPolicyRuleId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-InputObject] <AssignmentPolicyRule[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-InputObject] <AssignmentPolicyRule[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-AssignmentPolicyRuleName] <String>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerAssignmentPolicyRuleMetadata
   [-AssignmentPolicyRuleName] <String>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

The Set-BrokerAssignmentPolicyRuleMetadata cmdlet creates/updates metadata key-value

pairs for AssignmentPolicyRule. The AssignmentPolicyRule can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the AssignmentPolicyRule whose instance is pointed by $obj-Uid

```
Set-BrokerAssignmentPolicyRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the AssignmentPolicyRule in the site

```
Get-BrokerAssignmentPolicyRule | Set-BrokerAssignmentPolicyRuleMetadata
     -Name "MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the AssignmentPolicyRule in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerAssignmentPolicyRuleMetadata 'objname'
```

Parameters

-AssignmentPolicyRuleId

Specifies the AssignmentPolicyRule object whose Metadata is to be created/updated by ID.

Type: Int32

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the AssignmentPolicyRule objects whose Metadata is to be created/updated.

Type:	AssignmentPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-AssignmentPolicyRuleName

Specifies the AssignmentPolicyRule object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAssignmentPolicyRule

You can pipe the AssignmentPolicyRule to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerAssignmentPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerAssignmentPolicyRule object.

Related Links

Set-BrokerAutoTagRule

March 11, 2024

Changes an existing AutoTagRule.

Syntax

```
Set-BrokerAutoTagRule
    [-InputObject] <AutoTagRule[]>
    [-PassThru]
    [-Description <String>]
    [-ObjectType <String>]
    [-RuleText <String>]
    [-TagUid <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRule
    [-Name] <String>
    [-PassThru]
    [-Description <String>]
    [-ObjectType <String>]
    [-RuleText <String>]
    [-TagUid <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Changes the specified properties of an existing AutoTagRule.

Examples

EXAMPLE 1

Sets RandomAllocatedCatalogs'tag to MyExample2.

```
$tag = New-BrokerTag -Name MyExample2
Set-BrokerAutoTagRule -Name RandomAllocatedCatalogs -TagUid $tag.Uid
```


Parameters

-InputObject

Specifies the AutoTagRule object to change.

Type:	AutoTagRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the AutoTagRule to change.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Description

New Description that will be set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ObjectType

New ObjectType that will be set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RuleText

New RuleText that will be set.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-TagUid

New TagUid that will be set.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.AutoTagRule

AutoTagRule may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_AutoTagRule
- Get-BrokerAutoTagRule
- New-BrokerAutoTagRule
- Remove-BrokerAutoTagRule
- Rename-BrokerAutoTagRule

Set-BrokerAutoTagRuleMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for AutoTagRule

Syntax

```
Set-BrokerAutoTagRuleMetadata
   [-AutoTagRuleId] <Int32>
   [-PassThru]
   -Name <String>
```



```
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-AutoTagRuleId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-AutoTagRuleId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-InputObject] <AutoTagRule[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-InputObject] <AutoTagRule[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-AutoTagRuleName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerAutoTagRuleMetadata
    [-AutoTagRuleName] <String>
    [-PassThru]
    -Map <PSObject>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerAutoTagRuleMetadata cmdlet creates/updates metadata key-value

pairs for AutoTagRule. The AutoTagRule can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the AutoTagRule whose instance is pointed by $obj-Uid

```
Set-BrokerAutoTagRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the AutoTagRule in the site

```
Get-BrokerAutoTagRule | Set-BrokerAutoTagRuleMetadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the AutoTagRule in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerAutoTagRuleMetadata 'objname'
```

Parameters

-AutoTagRuleId

Specifies the AutoTagRule object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the AutoTagRule objects whose Metadata is to be created/updated.

Type:	AutoTagRule[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-AutoTagRuleName

Specifies the AutoTagRule object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a

series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerAutoTagRule

You can pipe the AutoTagRule to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerAutoTagRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerAutoTagRule object.

Related Links

Set-BrokerCatalog

March 11, 2024

Sets the properties of a catalog.

Syntax

```
Set-BrokerCatalog
   [-InputObject] <Catalog[]>
   [-PassThru]
   [-Description <String>]
   [-IsRemotePC <Boolean>]
   [-MinimumFunctionalLevel <FunctionalLevel>]
   [-ProvisioningSchemeId <Guid>]
   [-PvsAddress <String>]
   [-PvsDomain <String>]
   [-RemotePCHypervisorConnectionUid <Int32>]
   [-TimeZone <String>]
   [-ZoneUid <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerCatalog
   [-Name] <String>
   [-PassThru]
   [-Description <String>]
   [-IsRemotePC <Boolean>]
   [-MinimumFunctionalLevel <FunctionalLevel>]
   [-ProvisioningSchemeId <Guid>]
   [-PvsAddress <String>]
   [-PvsDomain <String>]
   [-RemotePCHypervisorConnectionUid <Int32>]
   [-TimeZone <String>]
   [-ZoneUid <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerCatalog cmdlet sets properties of a catalog or set of catalogs. The catalog can be specified by name, in which case only one catalog can be specified, or one or more catalog instances can

be passed to the command either by piping or by using the -InputObject parameter.

Examples

EXAMPLE 1

This example specifies a catalog by name and sets its description.

```
Set-BrokerCatalog -Name "MyCatalog" -Description "New Description"
```

EXAMPLE 2

This example sets the description for all catalogs with AllocationType 'Static'.

```
$permCatalogs = Get-BrokerCatalog -AllocationType Static
Set-BrokerCatalog -InputObject $permCatalogs -Description "Permanently
    assigned machines"
```

Parameters

-InputObject

Specifies the catalog objects to modify.

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Identifies the catalog to modify.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Supplies the new value of the Description property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsRemotePC

Supplies a new value for IsRemotePC.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- MachinesArePhysical is true.

IsRemotePC can only be set from true to false when no RemotePCAccount references this catalog, and when no Remote PC relationship exists between this catalog and a desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MinimumFunctionalLevel

The new minimum FunctionalLevel required for machines to work successfully in the catalog. If this is higher than the FunctionalLevel of any machines already in the catalog, they will immediately cease to function.

Valid values are L5, L7, L7_6, L7_7, L7_8, L7_9, L7_20, L7_25, L7_30, L7_34

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeId

Specifies the identity of the MCS provisioning scheme the catalog is associated with (can only be specified for new catalogs with a ProvisioningType of MCS; once set can never be changed).

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvsAddress

Supplies the new value of the PvsAddress property. Can only be set if CatalogKind is Pvs.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PvsDomain

Supplies the new value of the PvsDomain property. Can only be set if CatalogKind is Pvs.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemotePCHypervisorConnectionUid

Supplies the new hypervisor connection to use for powering on remote PCs in this catalog (only allowed when IsRemotePC is true); this will affect all machines already in the catalog as well as those created later.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TimeZone

The time zone in which this catalog's machines reside.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Supplies the Zone Uid associated with this catalog.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

You can pipe the catalogs to be modified to Set-BrokerCatalog.

Outputs

None or Citrix.Broker.Admin.SDK.Catalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Catalog object.

Notes

A catalog's Name property cannot be changed by Set-BrokerCatalog. To rename a catalog use Rename-BrokerCatalog.

Related Links

- about_Broker_RemotePC
- Get-BrokerCatalog
- New-BrokerCatalog
- Remove-BrokerCatalog
- Rename-BrokerCatalog
- Move-BrokerCatalog

Set-BrokerCatalogMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Catalog

Syntax

```
Set-BrokerCatalogMetadata
   [-CatalogId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
   [-CatalogId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
   [-CatalogId] <Int32>
   [-PassThru]
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
    [-InputObject] <Catalog[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
    [-InputObject] <Catalog[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
    [-CatalogName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerCatalogMetadata
    [-CatalogName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerCatalogMetadata cmdlet creates/updates metadata key-value pairs for Catalog. The Catalog can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Catalog whose instance is pointed by $obj-Uid

```
Set-BrokerCatalogMetadata -InputObject $obj-Uid -Name "MyMetadataName"
    -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Catalog in the site

```
Get-BrokerCatalog | Set-BrokerCatalogMetadata -Name "MyMetadataName" -
    Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Catalog in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerCatalogMetadata 'objname'
```

Parameters

-CatalogId

Specifies the Catalog object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Catalog objects whose Metadata is to be created/updated.

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-CatalogName

Specifies the Catalog object whose Metadata is to be created/updated by name.

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerCatalog

You can pipe the Catalog to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerCatalog

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerCatalog object.

Related Links

Set-BrokerCatalogRebootSchedule

March 11, 2024

Updates the values of one or more catalog reboot schedules.

Syntax

```
Set-BrokerCatalogRebootSchedule
    [-InputObject] <CatalogRebootSchedule[]>
    [-PassThru]
    [-Description <String>]
    [-Enabled <Boolean>]
```



```
    [-MaxOvertimeStartMins <Int32>]
    [-RebootDuration <Int32>]
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerCatalogRebootSchedule
    [-Name] <String>
    [-PassThru]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-MaxOvertimeStartMins <Int32>]
    [-RebootDuration <Int32>]
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerCatalogRebootSchedule cmdlet is used to alter the settings of an existing catalog reboot schedule.

Examples

EXAMPLE 1

Sets the catalog reboot schedule named Accounting to display a message with the title “WARNING: Reboot pending”and body “Save your work”ten minutes prior to rebooting each machine. The message is displayed in every user session on that machine.

```
Set-BrokerCatalogRebootSchedule -Name Accounting -WarningMessage "Save
    your work" -WarningDuration 10 -WarningTitle "WARNING: Reboot
    pending"
```


EXAMPLE 2

Sets all disabled reboot schedules to be enabled.

```
Get-BrokerCatalogRebootSchedule -Enabled $false | Set-
    BrokerCatalogRebootSchedule -Enabled $true
```

EXAMPLE 3

Sets the catalog reboot schedule for the reboot schedule having Uid 17 to display the message "Rebooting in %m% minutes."Fifteen, ten and five minutes prior to rebooting each machine, the message "Rebooting in %m% minutes."will be displayed in each user session with the pattern '%m%'replaced with the number of minutes until the reboot.

```
Set-BrokerCatalogRebootSchedule 17 -WarningMessage "Rebooting in %m%
    minutes." -WarningDuration 15 -WarningRepeatInterval 5
```

Parameters

-InputObject

The catalog reboot schedule to be modified.

Type:	CatalogRebootSchedule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the reboot schedule

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

An optional description for the reboot schedule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the reboot schedule is to be enabled or disabled.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDate

The date on which the schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.CatalogRebootSchedule

Catalog reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerCatalogRebootSchedule
- New-BrokerCatalogRebootSchedule
- Remove-BrokerCatalogRebootSchedule
- Rename-BrokerCatalogRebootSchedule

Set-BrokerConfiguration

March 11, 2024

This cmdlet is for internal use only

Syntax

```
Set-BrokerConfiguration
   [-Configuration <String>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet is for internal use only

Examples

Parameters

-Configuration

This parameter is for internal use only

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet does not accept input

Outputs

None

By default, this cmdlet returns no output.

Related Links

Set-BrokerConfigurationSlotMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for ConfigurationSlot

Syntax

```
Set-BrokerConfigurationSlotMetadata
   [-ConfigurationSlotId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
   [-ConfigurationSlotId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
   [-ConfigurationSlotId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
   [-InputObject] <ConfigurationSlot[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
   [-InputObject] <ConfigurationSlot[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
   [-ConfigurationSlotName] <String>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerConfigurationSlotMetadata
    [-ConfigurationSlotName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerConfigurationSlotMetadata cmdlet creates/updates metadata key-value pairs for ConfigurationSlot. The ConfigurationSlot can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the ConfigurationSlot whose instance is pointed by $obj-Uid

```
Set-BrokerConfigurationSlotMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the ConfigurationSlot in the site

```
Get-BrokerConfigurationSlot | Set-BrokerConfigurationSlotMetadata -Name
     "MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the ConfigurationSlot in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerConfigurationSlotMetadata 'objname'
```


Parameters

-ConfigurationSlotId

Specifies the ConfigurationSlot object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the ConfigurationSlot objects whose Metadata is to be created/updated.

Type:	ConfigurationSlot[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ConfigurationSlotName

Specifies the ConfigurationSlot object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerConfigurationSlot

You can pipe the ConfigurationSlot to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerConfigurationSlot

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerConfigurationSlot object.

Related Links

Set-BrokerControllerMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Controller

Syntax

```
Set-BrokerControllerMetadata
    [-ControllerId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-ControllerId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-ControllerId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-InputObject] <Controller[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-InputObject] <Controller[]>
    [-PassThru]
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-ControllerName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerControllerMetadata
    [-ControllerName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerControllerMetadata cmdlet creates/updates metadata key-value

pairs for Controller. The Controller can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Controller whose instance is pointed by $obj-Uid

```
Set-BrokerControllerMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Controller in the site

```
Get-BrokerController | Set-BrokerControllerMetadata -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Controller in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerControllerMetadata 'objname'
```

Parameters

-ControllerId

Specifies the Controller object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Controller objects whose Metadata is to be created/updated.

Type:	Controller[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ControllerName

Specifies the Controller object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerController

You can pipe the Controller to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerController

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerController object.

Related Links

Set-BrokerDBConnection

March 11, 2024

Configures a database connection for the Broker Service.

Syntax

```
Set-BrokerDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Broker Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-BrokerDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-BrokerDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Broker Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	3
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-BrokerDBConnection cmdlet returns an object describing the status of the Broker Service together with extra diagnostics information. Possible values are:

- OK:

The Broker Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Broker Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Broker Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Broker Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Broker Service currently in use is newer than, and incompatible with, the version of the Broker Service schema on the database. Upgrade the Broker Service to a more recent version.

- DBOlderVersionThanService:

The version of the Broker Service schema on the database is newer than, and incompatible with, the version of the Broker Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Broker Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Broker Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Broker Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerServiceStatus
- Get-BrokerDBConnection
- Test-BrokerDBConnection

Set-BrokerDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Broker Service.

Syntax

```
Set-BrokerDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerDBCredentials
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Broker Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Broker Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-BrokerDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-BrokerDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.


```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-BrokerDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-BrokerDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerDBSchema
- Set-BrokerDBConnection
- Get-Credential

Set-BrokerDesktopGroup

March 11, 2024

Adjusts the settings of a broker desktop group.

Syntax

```
Set-BrokerDesktopGroup
   [-InputObject] <DesktopGroup[]>
   [-PassThru]
   [-AllowReconnectInMaintenanceMode <Boolean>]
   [-AppDisks <Guid[]>]
   [-AppProtectionKeyLoggingRequired <Boolean>]
   [-AppProtectionScreenCaptureRequired <Boolean>]
   [-AutomaticPowerOnForAssigned <Boolean>]
   [-AutomaticPowerOnForAssignedDuringPeak <Boolean>]
   [-AutomaticRestartForUntaggedMachines <Boolean>]
   [-AutoscaleLogOffReminderEnabled <Boolean>]
   [-AutoscaleLogOffReminderIntervalSecondsOffPeak <Int32>]
   [-AutoscaleLogOffReminderIntervalSecondsPeak <Int32>]
   [-AutoscaleLogOffReminderMessage <String>]
   [-AutoscaleLogOffReminderTitle <String>]
   [-AutoscaleLogOffWarningMessage <String>]
```



```
    [-AutoscaleLogOffWarningTitle <String>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak <Int32>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak <Int32>]
    [-AutoscalingEnabled <Boolean>]
    [-ColorDepth <ColorDepth>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DisconnectOffPeakIdleSessionAfterSeconds <Int32>]
    [-DisconnectPeakIdleSessionAfterSeconds <Int32>]
    [-Enabled <Boolean>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IsRemotePC <Boolean>]
    [-LicenseModel <LicenseModel>]
    [-LogoffOffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-LogoffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-MachineCost <Decimal>]
    [-MachineLogOnType <MachineLogOnType>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-OffPeakBufferSizePercent <Int32>]
    [-OffPeakDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakDisconnectTimeout <Int32>]
    [-OffPeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakExtendedDisconnectTimeout <Int32>]
    [-OffPeakLogOffAction <SessionChangeHostingAction>]
    [-OffPeakLogOffTimeout <Int32>]
    [-PeakAutoscaleAssignedPowerOnIdleAction <String>]
    [-PeakAutoscaleAssignedPowerOnIdleTimeout <Int32>]
    [-PeakBufferSizePercent <Int32>]
    [-PeakDisconnectAction <SessionChangeHostingAction>]
    [-PeakDisconnectTimeout <Int32>]
    [-PeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-PeakExtendedDisconnectTimeout <Int32>]
    [-PeakLogOffAction <SessionChangeHostingAction>]
    [-PeakLogOffTimeout <Int32>]
    [-PolicySetGuid <Guid>]
    [-PowerOffDelay <Int32>]
    [-ProductCode <String>]
    [-ProtocolPriority <String[]>]
    [-PublishedName <String>]
    [-RequiredSleepCapability <String>]
    [-ResourceLeasingEnabled <Boolean>]
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak <Int32>]
    [-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak <Int32>]
    [-RestrictAutoscaleTagUid <Int32>]
    [-ReuseMachinesWithoutShutdownInOutage <Boolean>]
    [-SecureIcaRequired <Boolean>]
    [-SettlementPeriodBeforeAutoShutdown <TimeSpan>]
    [-SettlementPeriodBeforeUse <TimeSpan>]
    [-ShutdownDesktopsAfterUse <Boolean>]
    [-TimeZone <String>]
    [-TurnOnAddedMachine <Boolean>]
    [-UseVerticalScaling <Boolean>]
```



```
    [-ZonePreferences <ZonePreference[]>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerDesktopGroup
    [-Name] <String>
    [-PassThru]
    [-AllowReconnectInMaintenanceMode <Boolean>]
    [-AppDisks <Guid[]>]
    [-AppProtectionKeyLoggingRequired <Boolean>]
    [-AppProtectionScreenCaptureRequired <Boolean>]
    [-AutomaticPowerOnForAssigned <Boolean>]
    [-AutomaticPowerOnForAssignedDuringPeak <Boolean>]
    [-AutomaticRestartForUntaggedMachines <Boolean>]
    [-AutoscaleLogOffReminderEnabled <Boolean>]
    [-AutoscaleLogOffReminderIntervalSecondsOffPeak <Int32>]
    [-AutoscaleLogOffReminderIntervalSecondsPeak <Int32>]
    [-AutoscaleLogOffReminderMessage <String>]
    [-AutoscaleLogOffReminderTitle <String>]
    [-AutoscaleLogOffWarningMessage <String>]
    [-AutoscaleLogOffWarningTitle <String>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak <Int32>]
    [-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak <Int32>]
    [-AutoscalingEnabled <Boolean>]
    [-ColorDepth <ColorDepth>]
    [-DeliveryType <DeliveryType>]
    [-Description <String>]
    [-DisconnectOffPeakIdleSessionAfterSeconds <Int32>]
    [-DisconnectPeakIdleSessionAfterSeconds <Int32>]
    [-Enabled <Boolean>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IsRemotePC <Boolean>]
    [-LicenseModel <LicenseModel>]
    [-LogoffOffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-LogoffPeakDisconnectedSessionAfterSeconds <Int32>]
    [-MachineCost <Decimal>]
    [-MachineLogOnType <MachineLogOnType>]
    [-MinimumFunctionalLevel <FunctionalLevel>]
    [-OffPeakBufferSizePercent <Int32>]
    [-OffPeakDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakDisconnectTimeout <Int32>]
    [-OffPeakExtendedDisconnectAction <SessionChangeHostingAction>]
    [-OffPeakExtendedDisconnectTimeout <Int32>]
    [-OffPeakLogOffAction <SessionChangeHostingAction>]
    [-OffPeakLogOffTimeout <Int32>]
    [-PeakAutoscaleAssignedPowerOnIdleAction <String>]
    [-PeakAutoscaleAssignedPowerOnIdleTimeout <Int32>]
    [-PeakBufferSizePercent <Int32>]
    [-PeakDisconnectAction <SessionChangeHostingAction>]
    [-PeakDisconnectTimeout <Int32>]
    [-PeakExtendedDisconnectAction <SessionChangeHostingAction>]
```



```
49      [-PeakExtendedDisconnectTimeout <Int32>]
50      [-PeakLogOffAction <SessionChangeHostingAction>]
51      [-PeakLogOffTimeout <Int32>]
52      [-PolicySetGuid <Guid>]
53      [-PowerOffDelay <Int32>]
54      [-ProductCode <String>]
55      [-ProtocolPriority <String[]>]
56      [-PublishedName <String>]
57      [-RequiredSleepCapability <String>]
58      [-ResourceLeasingEnabled <Boolean>]
59      [-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak <Int32>]
60      [-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak <Int32>]
61      [-RestrictAutoscaleTagUid <Int32>]
62      [-ReuseMachinesWithoutShutdownInOutage <Boolean>]
63      [-SecureIcaRequired <Boolean>]
64      [-SettlementPeriodBeforeAutoShutdown <TimeSpan>]
65      [-SettlementPeriodBeforeUse <TimeSpan>]
66      [-ShutdownDesktopsAfterUse <Boolean>]
67      [-TimeZone <String>]
68      [-TurnOnAddedMachine <Boolean>]
69      [-UseVerticalScaling <Boolean>]
70      [-ZonePreferences <ZonePreference[]>]
71      [-LoggingId <Guid>]
72      [<CitrixCommonParameters>]
73      [<CommonParameters>]
```

Description

The Set-BrokerDesktopGroup cmdlet is used to disable or enable an existing broker desktop group or to alter its settings.

Examples

EXAMPLE 1

Sets all desktop groups with names starting “EMEA”into maintenance mode, returning the set of desktop groups.

```
1  Set-BrokerDesktopGroup EMEA* -InMaintenanceMode $true -PassThru
```

EXAMPLE 2

Disable all desktop groups that are in maintenance mode.

```
1  Get-BrokerDesktopGroup -InMaintenanceMode $true | Set-
       BrokerDesktopGroup -Enabled $false
```


Parameters

-InputObject

Specifies the desktop groups to adjust.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the desktop groups to adjust, based on their Name property.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AllowReconnectInMaintenanceMode

Whether the maintenance mode allows session reconnect to various Session support types.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppDisks

Specifies the application disks to be used by machines in the desktop group.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionKeyLoggingRequired

Specifies whether key logging app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AppProtectionScreenCaptureRequired

Specifies whether screen capture app protection is required.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticPowerOnForAssigned

Specifies whether assigned desktops in the desktop group should be automatically started at the start of peak time periods. Only relevant for groups whose DesktopKind is Private.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticPowerOnForAssignedDuringPeak

Specifies whether assigned desktops in the desktop group should be automatically started throughout peak time periods. Only relevant for groups whose DesktopKind is Private and which have AutomaticPowerOnForAssigned set to true.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutomaticRestartForUntaggedMachines

Indicates whether untagged single-session machines belonging to a desktop group configured for restrict Autoscale and shutdown after use should restart after untainting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderEnabled

Gets the value indicating whether the warning messages should be sent on an interval to nudge a logoff should be sent on an interval when autoscale is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsOffPeak

Gets the interval in seconds which represents the time interval at which messages are sent to the user during off peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderIntervalSecondsPeak

Gets the interval in seconds which represents the time interval at which messages are sent to the user during peak time when autoscale is enabled. This message will nudge users to log off instead of forcibly logging them off

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderMessage

Specifies the notification message to display to users in active sessions belonging to machines needed by Autoscale for shutdown

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffReminderTitle

Specifies the notification message dialog title displayed when Autoscale issues a logoff reminder request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffWarningMessage

Specifies the warning message to display to users in active sessions prior to Autoscale issuing a logoff request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleLogOffWarningTitle

Specifies the warning message dialog title displayed prior to Autoscale issuing a logoff request.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleMaxSecondsBeforeForcedLogOffDuringOffPeak

Specifies the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group during off-peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscaleMaxSecondsBeforeForcedLogOffDuringPeak

Specifies the minimum seconds that need to elapse before Autoscale logs off the active sessions on the draining machines belonging to the delivery group peak time. This property will override the power-off delay behavior if it is configured to a value greater than zero.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoscalingEnabled

Specifies whether machines in this desktop group can be Autoscaled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Specifies the color depth that the ICA session should use for desktops in this group. Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeliveryType

Specifies whether desktops, applications, or both, can be delivered from machines contained within the desktop group. Desktop groups with a DesktopKind of Private cannot be used to deliver both desktops and applications.

When changing the delivery type to desktops only, there must be no remaining desktop-hosted applications associated with the group, or application-specific assignment/entitlement policy rules for the group.

When changing the delivery type to applications only, there must be no remaining client-hosted applications associated with the group, or desktop-specific assignment/entitlement policy rules for the group.

Valid values are DesktopsOnly, AppsOnly, and DesktopsAndApps.

Type:	DeliveryType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

A description for this desktop group useful for administrators of the site.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisconnectOffPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during off-peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisconnectPeakIdleSessionAfterSeconds

Specifies the time in seconds after which an idle session belonging to the delivery group is disconnected during peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Whether the desktop group should be in the enabled state; disabled desktop groups do not appear to users.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

The UID of the broker icon to be displayed to users for their desktop(s) in this desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-InMaintenanceMode

Whether the desktop should be put into maintenance mode; a desktop group in maintenance mode will not allow users to connect or reconnect to their desktops.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsRemotePC

Supplies a new value for IsRemotePC.

IsRemotePC can only be enabled when:

- SessionSupport is SingleSession
- DeliveryType is DesktopsOnly
- DesktopKind is Private

IsRemotePC can be switched from true to false only if no RemotePC relationship exists between a catalog and this desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseModel

The license model for this desktop group. If none is specified, then the site-wide license model is used.

Type:	LicenseModel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffOffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during off-peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffPeakDisconnectedSessionAfterSeconds

Specifies the time in seconds after which a disconnected session belonging to the delivery group is terminated during peak time.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MachineCost

The per-hour instance cost of machines in this desktop group.

Type:	Decimal
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineLogOnType

Specifies the login type for the desktop group. Values can be:

- ActiveDirectory - Perform Active Directory login on the machine.
- LocalMappedAccount - Perform local mapped account login on the machine.

Type:	MachineLogOnType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MinimumFunctionalLevel

The new minimum FunctionalLevel required for machines to work successfully in the desktop group. If this is higher than the FunctionalLevel of any machines already in the desktop group, they will immediately cease to function.

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakBufferSizePercent

The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available outside peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakDisconnectAction

The action to be performed after a configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakDisconnectTimeout

The number of minutes before the configured action should be performed after a user session disconnects outside peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakExtendedDisconnectAction

The action to be performed after a second configurable period of a user session disconnecting outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakExtendedDisconnectTimeout

The number of minutes before the second configured action should be performed after a user session disconnects outside peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakLogOffAction

The action to be performed after a configurable period of a user session ending outside peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OffPeakLogOffTimeout

The number of minutes before the configured action should be performed after a user session ends outside peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleAction

The action to be performed on an assigned machine started by Autoscale if that machine then remains unused for a defined period of time.

Type:	String
Accepted values:	Nothing, Shutdown, Suspend
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakAutoscaleAssignedPowerOnIdleTimeout

The number of minutes before the configured action is performed on an assigned machine previously started by autoscale that subsequently remains unused.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakBufferSizePercent

The percentage of single-session machines that are kept available in an idle state, or for multi-session machines, the percentage of the total load capacity to be kept available in peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakDisconnectAction

The action to be performed after a configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakDisconnectTimeout

The number of minutes before the configured action should be performed after a user session disconnects in peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakExtendedDisconnectAction

The action to be performed after a second configurable period of a user session disconnecting in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PeakExtendedDisconnectTimeout

The number of minutes before the second configured action should be performed after a user session disconnects in peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakLogOffAction

The action to be performed after a configurable period of a user session ending in peak hours. Possible values are Nothing, Suspend, or Shutdown.

Type:	SessionChangeHostingAction
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakLogOffTimeout

The number of minutes before the configured action should be performed after a user session ends in peak hours.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicySetGuid

The policy set assigned to this delivery group.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PowerOffDelay

The number of minutes following a power-on operation that Auto Scale will wait before attempting to power off that same machine. Possible values are in the range 0 - 60.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductCode

The licensing product code for this desktop group. If none is specified, then the site-wide product code is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProtocolPriority

A list of protocol names in the order in which they should be attempted for use during connection.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

The name that will be displayed to users for their desktop(s) in this desktop group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RequiredSleepCapability

The sleep capability of this desktop group. Possible values are None, or Suspend.

Type:	String
Accepted values:	None, Suspend
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceLeasingEnabled

Indicates if this desktop group is enabled for resource leasing.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringOffPeak

This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages ‘tagged’machines, as per policy, in off-peak. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down ‘tagged’machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for desktop groups configured for Restrict Autoscaling.

Type:	Int32

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleMinIdleUntaggedPercentDuringPeak

This indicates the percentage that the number of untagged single-session machines in an idle state, or for multi-session machines, the untagged available load capacity must fall below before Autoscale powers on and manages 'tagged' machines, as per policy, in peak time. If the number of untagged machines in an idle state, or the untagged available load capacity goes above this threshold value, Autoscale will attempt to shut down 'tagged' machines. Possible values are in the range -1 - 100, where -1 (default) disables this behavior. This is only relevant for desktop groups configured for Restrict Autoscaling.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictAutoscaleTagUid

If set to a Tag UID, only machines that have this tag will be auto scaled.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ReuseMachinesWithoutShutdownInOutage

Specifies whether power cycle operations are required during outage for pooled machines.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Whether HDX connections to desktops in the new desktop group require the use of a secure protocol.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettlementPeriodBeforeAutoShutdown

Time after a session ends during which automatic shutdown requests (for example, shutdown after use, idle pool management) are deferred. Any outstanding shutdown request takes effect after the settlement period expires. This is typically used to configure time to allow logoff scripts to complete.

Type:	TimeSpan

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettlementPeriodBeforeUse

Idle period before a machine can be selected to host a new session after registration or the end of a previous session. This is typically used to allow a machine to become idle following processing associated with start-up or logoff actions. A machine may still be selected during the idle period if no other machine is available for immediate use.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ShutdownDesktopsAfterUse

Whether desktops in this desktop group should be automatically shut down when each user session completes (only relevant to power-managed desktops).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TimeZone

The time zone in which this desktop group's machines reside.

The time zone must be specified for any of the group's automatic power management settings to take effect. Automatic power management operations include pool management (power time schemes), reboot schedules, session disconnect and logoff actions, and powering on assigned machines etc.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TurnOnAddedMachine

This flag specifies whether the Broker Service should attempt to power on machines when they are added to the desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseVerticalScaling

Determines whether to use vertical scaling when considering RDS machines in the desktop group for launches. Vertical scaling would saturate machines in the current pool rather than send sessions to the least loaded machines. This would be a trade in performance vs. cost, where vertical scaling would be less costly.

With the default value ($null), the site-wide value will be inherted. A value of $false indicates that horizontal scaling should be used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZonePreferences

Ordered list of zone preferences to be applied when launching resources from this desktop group. Valid zone preference values are UserLocation, UserHome, UserHomeOnly and ApplicationHome.

The list can have zero or more entries subject to the following restrictions: Zone preferences can only be applied to groups having a DesktopKind of Shared; the same zone preference value cannot occur in the list more than once; the UserHome and UserHomeOnly values are mutually exclusive and cannot both appear in the list.

Type:	ZonePreference[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

You can pipe the desktop groups to be adjusted to Set-BrokerDesktopGroup.

Outputs

None or Citrix.Broker.Admin.SDK.DesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.DesktopGroup object.

Related Links

- about_Broker_Desktops
- about_Broker_RemotePC
- Get-BrokerDesktopGroup
- New-BrokerDesktopGroup
- Rename-BrokerDesktopGroup
- Move-BrokerDesktopGroup
- Remove-BrokerDesktopGroup

Set-BrokerDesktopGroupMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for DesktopGroup

Syntax

```
Set-BrokerDesktopGroupMetadata
   [-DesktopGroupId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
   [-DesktopGroupId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
   [-DesktopGroupId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
   [-InputObject] <DesktopGroup[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
   [-InputObject] <DesktopGroup[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
    [-DesktopGroupName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerDesktopGroupMetadata
    [-DesktopGroupName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerDesktopGroupMetadata cmdlet creates/updates metadata key-value pairs for DesktopGroup. The DesktopGroup can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the DesktopGroup whose instance is pointed by $obj-Uid

```
Set-BrokerDesktopGroupMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the DesktopGroup in the site

```
Get-BrokerDesktopGroup | Set-BrokerDesktopGroupMetadata -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the DesktopGroup in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerDesktopGroupMetadata 'objname'
```

Parameters

-DesktopGroupId

Specifies the DesktopGroup object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the DesktopGroup objects whose Metadata is to be created/updated.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

Specifies the DesktopGroup object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerDesktopGroup

You can pipe the DesktopGroup to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerDesktopGroup

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerDesktopGroup object.

Related Links

Set-BrokerDesktopGroupWebhook

March 11, 2024

Adjusts the settings of a webhook for a desktop group

Syntax

```
Set-BrokerDesktopGroupWebhook
   [-InputObject] <DesktopGroupWebhook[]>
   [-PassThru]
   [-Address <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet is used to alter the settings of a webhook for a desktop group

Examples

EXAMPLE 1

Changes the URL of the webhook for the desktop group with Uid 1


```
Get-BrokerDesktopGroupWebhook -DesktopGroupUid 1 | Set-
    BrokerDesktopGroupWebhook -Address "http://citrix.com/example2"
```

Parameters

-InputObject

The webhook object to alter the settings.

Type:	DesktopGroupWebhook[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Address

Specified the new URL for the webhook.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroupWebhook

You can pipe webhooks to this cmdlet

Outputs

None or Citrix.Broker.Admin.SDK.DesktopGroupWebhook

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.DesktopGroupWebhook object.

Related Links

- Get-BrokerDesktopGroup

Set-BrokerEntitlementPolicyRule

March 11, 2024

Modifies an existing desktop rule in the site's entitlement policy.

Syntax

```
Set-BrokerEntitlementPolicyRule
   [-InputObject] <EntitlementPolicyRule[]>
   [-PassThru]
   [-AddExcludedUsers <User[]>]
   [-AddIncludedUsers <User[]>]
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-ExcludedUserFilterEnabled <Boolean>]
   [-ExcludedUsers <User[]>]
   [-IconUid <Int32>]
   [-IncludedUserFilterEnabled <Boolean>]
   [-IncludedUsers <User[]>]
   [-LeasingBehavior <LeasingBehavior>]
   [-MaxPerEntitlementInstances <Int32>]
   [-PublishedName <String>]
   [-RemoveExcludedUsers <User[]>]
   [-RemoveIncludedUsers <User[]>]
   [-RestrictToTag <String>]
   [-SecureIcaRequired <Boolean>]
   [-SessionReconnection <SessionReconnection>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerEntitlementPolicyRule
    [-Name] <String>
    [-PassThru]
    [-AddExcludedUsers <User[]>]
    [-AddIncludedUsers <User[]>]
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-ExcludedUserFilterEnabled <Boolean>]
    [-ExcludedUsers <User[]>]
    [-IconUid <Int32>]
    [-IncludedUserFilterEnabled <Boolean>]
    [-IncludedUsers <User[]>]
    [-LeasingBehavior <LeasingBehavior>]
    [-MaxPerEntitlementInstances <Int32>]
    [-PublishedName <String>]
    [-RemoveExcludedUsers <User[]>]
    [-RemoveIncludedUsers <User[]>]
    [-RestrictToTag <String>]
    [-SecureIcaRequired <Boolean>]
    [-SessionReconnection <SessionReconnection>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerEntitlementPolicyRule cmdlet modifies an existing desktop rule in the site's entitlement policy.

A desktop rule in the entitlement policy defines the users who are allowed per-session access to a machine from the rule's associated desktop group to run a full desktop session.

Changing a rule does not affect existing sessions launched using the rule, but if the change removes an entitlement to a machine that was previously granted, users may be unable to reconnect to a disconnected session on that machine.

Examples

EXAMPLE 1

Adds the user group OFFICE\contractors to the Temp Workers desktop rule of the entitlement policy. This grants all members of that group an entitlement to a desktop session in the rule's associated desktop group. The session properties of the desktops obtained using the rule are determined by the rule's other properties.


```
Set-BrokerEntitlementPolicyRule 'Temp Workers' -AddIncludedUsers office
    \contractors
```

EXAMPLE 2

Disables the Temp Workers desktop rule in the entitlement policy. This prevents further desktop sessions being launched using this rule until it is re-enabled. However, access to existing desktop sessions is not affected.

```
Set-BrokerEntitlementPolicyRule 'Temp Workers' -Enabled $false
```

Parameters

-InputObject

The desktop rule in the entitlement policy to be modified.

Type:	EntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the desktop rule in the entitlement policy to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddExcludedUsers

Adds the specified users to the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AddIncludedUsers

Adds the specified users to the included users filter of the rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Changes the color depth of any desktop sessions launched by a user from this entitlement. Existing sessions are not affected.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the desktop rule. The text may be visible to the end user, for example, as a tooltip associated with the desktop entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Enables or disables the desktop rule. A disabled rule is ignored when evaluating the site's entitlement policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUserFilterEnabled

Enables or disables the excluded users filter. If the filter is disabled then any user entries in the filter are ignored when entitlement policy rules are evaluated.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedUsers

Changes the excluded users filter of the desktop rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

This can be used to exclude users or groups who would otherwise gain access by groups specified in the included users filter.

Type:	User[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Changes the icon (identified by its unique ID) for the published desktop entitlement as seen by the user.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUserFilterEnabled

Enables or disables the included users filter. If the filter is disabled then any user who satisfies the requirements of the access policy is implicitly granted an entitlement to a desktop session by the rule.

Users who would be implicitly granted access when the filter is disabled can still be explicitly denied access using the excluded users filter.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludedUsers

Changes the included users filter of the desktop rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

If a user appears explicitly in the excluded users filter of the rule or is a member of a group that appears in the excluded users filter, no entitlement is granted whether or not the user appears in the included users filter.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LeasingBehavior

Defines the desired connection leasing behavior applied to sessions launched using this entitlement. Possible values are:

Allowed and Disallowed.

The Allowed value indicates that connection leasing should behave normally. The Disallowed value prevents users

from launching or reconnecting to sessions using this entitlement while connection leasing is active (typically during a database outage).

Type:	LeasingBehavior
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxPerEntitlementInstances

Maximum allowed concurrently running instances of the desktop associated with this entitlement in the site . A value of zero allows unlimited usage.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Changes the name of the published desktop entitlement as seen by the user.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveExcludedUsers

Removes the specified users from the excluded users filter of the rule, that is, the users and groups who are explicitly denied an entitlement to a desktop session from this rule.

See the ExcludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RemoveIncludedUsers

Removes the specified users from the included users filter of the desktop rule, that is, the users and groups who are granted an entitlement to a desktop session by the rule.

See the IncludedUsers parameter for more information.

Type:	User[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

Optional tag that may be used further to restrict which machines may be made accessible to a user by an entitlement policy rule. A machine may be made accessible by an entitlement policy rule only if either the rule has no tag restriction or the rule does have a tag restriction and the machine is tagged with the same tag.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Changes whether the desktop rule requires the SecureICA protocol for desktop sessions launched using the entitlement.

A null value indicates that the equivalent setting from the rule's desktop group is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionReconnection

Defines reconnection (roaming) behavior for sessions launched using this rule. Possible values are:

Always, DisconnectedOnly, and SameEndpointOnly.

Type:	SessionReconnection
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.EntitlementPolicyRule

The desktop rule in the entitlement policy to be modified.

Outputs

None or Citrix.Broker.Admin.SDK.EntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.EntitlementPolicyRule object.

Related Links

- about_Broker_Policies
- about_Broker_AccessPolicy
- about_Broker_EntitlementPolicy
- about_Broker_ErrorHandling

- New-BrokerEntitlementPolicyRule
- Get-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule
- Remove-BrokerEntitlementPolicyRule

Set-BrokerEntitlementPolicyRuleMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for EntitlementPolicyRule

Syntax

```
Set-BrokerEntitlementPolicyRuleMetadata
   [-EntitlementPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
   [-EntitlementPolicyRuleId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
   [-EntitlementPolicyRuleId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
   [-InputObject] <EntitlementPolicyRule[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
    [-InputObject] <EntitlementPolicyRule[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
    [-EntitlementPolicyRuleName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerEntitlementPolicyRuleMetadata
    [-EntitlementPolicyRuleName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerEntitlementPolicyRuleMetadata cmdlet creates/updates metadata key-value pairs for EntitlementPolicyRule. The EntitlementPolicyRule can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the EntitlementPolicyRule whose instance is pointed by $obj-Uid

```
Set-BrokerEntitlementPolicyRuleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the EntitlementPolicyRule in the site

```
Get-BrokerEntitlementPolicyRule | Set-
    BrokerEntitlementPolicyRuleMetadata -Name "MyMetadataName" -Value "
    1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the EntitlementPolicyRule in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerEntitlementPolicyRuleMetadata 'objname'
```

Parameters

-EntitlementPolicyRuleId

Specifies the EntitlementPolicyRule object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the EntitlementPolicyRule objects whose Metadata is to be created/updated.

Type:	EntitlementPolicyRule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-EntitlementPolicyRuleName

Specifies the EntitlementPolicyRule object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule

You can pipe the EntitlementPolicyRule to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerEntitlementPolicyRule object.

Related Links

Set-BrokerGpoFilter

March 11, 2024

Changes the data of an existing GPO filter.

Syntax

```
Set-BrokerGpoFilter
    [-InputObject] <GpoFilter[]>
    [-PassThru]
    [-FilterData <String>]
    [-IsAllowed <Boolean>]
    [-IsEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Changes the data of an existing GPO filter. No other properties of the filter can be modified.

Examples

EXAMPLE 1

Changes the data of a filter to 'Client2'.

```
Set-BrokerGpoFilter ([Guid]"12345678-...") -FilterData 'Client2'
```

Parameters

-InputObject

Specifies the GPO filter object to set.

Type:	GpoFilter[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterData

The filter data. Ignored if the filter is a BranchRepeater filter.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsAllowed

Specifies if policy is allowed if the filter condition is true.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsEnabled

Specifies if the filter is enabled or not.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoFilter

GPO filters may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoFilter
- New-BrokerGpoFilter
- Remove-BrokerGpoFilter

Set-BrokerGpoPolicy

March 11, 2024

Changes the properties of a GPO policy.

Syntax

```
Set-BrokerGpoPolicy
    [-InputObject] <GpoPolicy[]>
    [-PassThru]
    [-Description <String>]
    [-IsEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerGpoPolicy
    [-Name] <String>
    [-PassThru]
    [-Description <String>]
    [-IsEnabled <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerGpoPolicy cmdlet is used to disable or enable an existing GPO policy or to alter its properties.

Examples

EXAMPLE 1

Enable all policies with names containing the word 'test'.

```
Get-BrokerGpoPolicy -Name '*test*' | Set-BrokerGpoPolicy -IsEnabled
    $true
```

Parameters

-InputObject

Specifies the GPO policy object to set.

Type:	GpoPolicy[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies a new name for the policy.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Specifies a new description for the policy.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsEnabled

Enable or disable the policy.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoPolicy

You can pipe GPO policies to be adjusted to Set-BrokerGpoPolicy.

Outputs

None or Citrix.Broker.Admin.SDK.GpoPolicy

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.GpoPolicy object.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoPolicy
- New-BrokerGpoPolicy
- Remove-BrokerGpoPolicy

Set-BrokerGpoPolicyPriorities

March 11, 2024

Changes the priorities of policies in a GPO policy set.

Syntax

```
Set-BrokerGpoPolicyPriorities
    [-PolicySetGuid <Guid>]
    [-PolicyGuids] <Guid[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerGpoPolicyPriorities cmdlet is used to change the priorities of the policies in a policy set. The input must be an array of GUIDs and it must be the complete set of all policy GUIDs in the policy set and there must be no duplications of GUIDs in the input.

Examples

EXAMPLE 1

Set the new policy priority order.

```
Set-BrokerGpoPolicyPriorities -PolicyGuids @("abcdef12-...", "
    12345678-...")
```


Parameters

-PolicyGuids

The list GUIDS of all the policies in the policy set. The order of the GUIDs indicate the new priority oder.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PolicySetGuid

The GUID of the policy set that contains the policies. If this is not specified, this means the default site policy set.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

This cmdlet does not generate any output.

Related Links

- about_Broker_GroupPolicy

Set-BrokerGpoPolicySet

March 11, 2024

Changes the data of an existing GPO policy set.

Syntax

```
Set-BrokerGpoPolicySet
   [-InputObject] <GpoPolicySet[]>
   [-PassThru]
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerGpoPolicySet
   [-Name] <String>
   [-PassThru]
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Changes the last convert mark or the description of an existing GPO policy set.

Examples

EXAMPLE 1

Changes the description of the policy set to ‘my policy set’.

```
Set-BrokerGpoPolicySet ([Guid]"12345678-...") -Description 'my policy
   set'
```

Parameters

-InputObject

Specifies the GPO policy set object to set.

Type:	GpoPolicySet[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the policy set. It must be unique among all policy sets.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

A short description of the policy set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoPolicySet

GPO policy sets may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoPolicySet
- New-BrokerGpoPolicySet
- Remove-BrokerGpoPolicySet

Set-BrokerGpoSetting

March 11, 2024

Changes the state and/or value of an existing GPO setting.

Syntax

```
Set-BrokerGpoSetting
    [-InputObject] <GpoSetting[]>
    [-PassThru]
    [-SettingValue <String>]
    [-UseDefault <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerGpoSetting
    [-SettingName] <String>
    [-PassThru]
    [-SettingValue <String>]
    [-UseDefault <Boolean>]
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Changes the state and/or value of an existing GPO setting.

Examples

EXAMPLE 1

Changes the state of the setting to 2 (disabled).

```
Set-BrokerGpoSetting ([Guid]"12345678-...") -UseDefault $true
```

Parameters

-InputObject

Specifies the GPO setting object to set.

Type:	GpoSetting[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SettingName

Specifies the GPO setting object to set.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettingValue

The setting value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseDefault

Indicate if default value of setting is used. Ignored if setting is Boolean.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.GpoSetting

GPO settings may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_GroupPolicy
- Get-BrokerGpoSetting
- New-BrokerGpoSetting
- Remove-BrokerGpoSetting

Set-BrokerHostingPowerAction

March 11, 2024

Changes the priority of one or more pending power actions.

Syntax

```
Set-BrokerHostingPowerAction
    [-InputObject] <HostingPowerAction[]>
    [-PassThru]
    [-ActualPriority <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerAction
    [-MachineName] <String>
    [-PassThru]
    [-ActualPriority <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
7     [<CommonParameters>]
```

Description

The Set-BrokerHostingPowerAction cmdlet modifies an existing power action in the site's power action queue. The only property of power actions you can change, is the current priority of the action.

For a detailed description of the queuing mechanism, see 'help about_Broker_PowerManagement' .

Examples

EXAMPLE 1

Sets the current priority of actions for the machine called 'XD_VDA1'to 25. Numerically lower priority values indicate more important actions that will be processed in preference to actions with numerically higher priority settings.

```
1  Set-BrokerHostingPowerAction -MachineName 'XD_VDA1' -ActualPriority 25
```

Parameters

-InputObject

The power action whose priority is to be changed.

Type:	HostingPowerAction[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Changes the priority of actions that are for machines whose name (of the form domain\machine) matches the specified string.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ActualPriority

Specifies a new priority value for the action in the queue.

This priority is the current action priority; the 'base'or original priority for actions cannot be altered. Numerically lower priority values indicate more important actions that are processed in preference to actions with numerically higher priority settings.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.HostingPowerAction

The power action whose priority is to be changed.

Outputs

None or Citrix.Broker.Admin.SDK.HostingPowerAction

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.HostingPowerAction object.

Related Links

- about_Broker_PowerManagement
- Get-BrokerHostingPowerAction
- New-BrokerHostingPowerAction
- Remove-BrokerHostingPowerAction

Set-BrokerHostingPowerActionMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for HostingPowerAction

Syntax

```
Set-BrokerHostingPowerActionMetadata
    [-HostingPowerActionId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-HostingPowerActionId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-HostingPowerActionId] <Int64>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-InputObject] <HostingPowerAction[]>
    [-PassThru]
    -Name <String>
```



```
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-InputObject] <HostingPowerAction[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-HostingPowerActionName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHostingPowerActionMetadata
    [-HostingPowerActionName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerHostingPowerActionMetadata cmdlet creates/updates metadata key-value pairs for HostingPowerAction. The HostingPowerAction can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the HostingPowerAction whose instance is pointed by $obj-Uid

```
Set-BrokerHostingPowerActionMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the HostingPowerAction in the site

```
Get-BrokerHostingPowerAction | Set-BrokerHostingPowerActionMetadata -
    Name "MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the HostingPowerAction in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerHostingPowerActionMetadata 'objname'
```

Parameters

-HostingPowerActionId

Specifies the HostingPowerAction object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the HostingPowerAction objects whose Metadata is to be created/updated.

Type:	HostingPowerAction[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-HostingPowerActionName

Specifies the HostingPowerAction object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHostingPowerAction

You can pipe the HostingPowerAction to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerHostingPowerAction

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerHostingPowerAction object.

Related Links

Set-BrokerHypervisorAlertMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for HypervisorAlert

Syntax

```
Set-BrokerHypervisorAlertMetadata
   [-HypervisorAlertId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerHypervisorAlertMetadata
   [-HypervisorAlertId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerHypervisorAlertMetadata
   [-HypervisorAlertId] <Int64>
```



```
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorAlertMetadata
    [-InputObject] <HypervisorAlert[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorAlertMetadata
    [-InputObject] <HypervisorAlert[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerHypervisorAlertMetadata cmdlet creates/updates metadata key-value pairs for HypervisorAlert. The HypervisorAlert can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the HypervisorAlert whose instance is pointed by $obj-Uid

```
Set-BrokerHypervisorAlertMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the HypervisorAlert in the site

```
Get-BrokerHypervisorAlert | Set-BrokerHypervisorAlertMetadata -Name "
    MyMetadataName" -Value "1234"
```


Parameters

-HypervisorAlertId

Specifies the HypervisorAlert object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the HypervisorAlert objects whose Metadata is to be created/updated.

Type:	HypervisorAlert[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHypervisorAlert

You can pipe the HypervisorAlert to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerHypervisorAlert

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerHypervisorAlert object.

Related Links

Set-BrokerHypervisorConnection

March 11, 2024

Sets the properties of a hypervisor connection.

Syntax

```
Set-BrokerHypervisorConnection
   [-InputObject] <HypervisorConnection[]>
   [-PassThru]
   [-PreferredController <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerHypervisorConnection
   [-Name] <String>
   [-PassThru]
   [-PreferredController <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerHypervisorConnection cmdlet sets the properties on a hypervisor connection.

Examples

EXAMPLE 1

Updates the preferred controller for a hypervisor connection.

```
$hvConn = Get-BrokerHypervisorConnection -PreferredController "
   oldController" -Name "Xen Server Connection"
Set-BrokerHypervisorConnection -InputObject $hvConn -
   PreferredController "newController"
```

Parameters

-InputObject

Specifies the hypervisor connection object to adjust.

Type:	HypervisorConnection[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the hypervisor connection object to adjust.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredController

Supplies the new value of the PreferredController property.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.HypervisorConnection

You can pipe the hypervisor connection to be modified to Set-BrokerHypervisorConnection.

Outputs

None or Citrix.Broker.Admin.SDK.HypervisorConnection

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.HypervisorConnection object.

Related Links

- Get-BrokerHypervisorConnection
- New-BrokerHypervisorConnection
- Remove-BrokerHypervisorConnection

Set-BrokerHypervisorConnectionMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for HypervisorConnection

Syntax

```
Set-BrokerHypervisorConnectionMetadata
   [-HypervisorConnectionId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
   [-HypervisorConnectionId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
    [-HypervisorConnectionId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
    [-InputObject] <HypervisorConnection[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
    [-InputObject] <HypervisorConnection[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
    [-HypervisorConnectionName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerHypervisorConnectionMetadata
    [-HypervisorConnectionName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerHypervisorConnectionMetadata cmdlet creates/updates metadata key-value

pairs for HypervisorConnection. The HypervisorConnection can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the HypervisorConnection whose instance is pointed by $obj-Uid

```
Set-BrokerHypervisorConnectionMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the HypervisorConnection in the site

```
Get-BrokerHypervisorConnection | Set-BrokerHypervisorConnectionMetadata
     -Name "MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the HypervisorConnection in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerHypervisorConnectionMetadata 'objname'
```

Parameters

-HypervisorConnectionId

Specifies the HypervisorConnection object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the HypervisorConnection objects whose Metadata is to be created/updated.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-HypervisorConnectionName

Specifies the HypervisorConnection object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerHypervisorConnection

You can pipe the HypervisorConnection to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerHypervisorConnection

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerHypervisorConnection object.

Related Links

Set-BrokerIconMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Icon

Syntax

```
Set-BrokerIconMetadata
    [-IconId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerIconMetadata
    [-IconId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerIconMetadata
    [-IconId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerIconMetadata
    [-InputObject] <Icon[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerIconMetadata
    [-InputObject] <Icon[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerIconMetadata cmdlet creates/updates metadata key-value pairs for Icon. The Icon can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Icon whose instance is pointed by $obj-Uid

```
Set-BrokerIconMetadata -InputObject $obj-Uid -Name "MyMetadataName" -
    Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Icon in the site

```
Get-BrokerIcon | Set-BrokerIconMetadata -Name "MyMetadataName" -Value "
    1234"
```

Parameters

-IconId

Specifies the Icon object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Icon objects whose Metadata is to be created/updated.

Type:	Icon[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerIcon

You can pipe the Icon to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerIcon

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerIcon object.

Related Links

Set-BrokerLeaseMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Lease

Syntax

```
Set-BrokerLeaseMetadata
   [-LeaseId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
   [-LeaseId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
   [-LeaseId] <Int64>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
   [-InputObject] <Lease[]>
```



```
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
    [-InputObject] <Lease[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
    [-LeaseName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerLeaseMetadata
    [-LeaseName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerLeaseMetadata cmdlet creates/updates metadata key-value

pairs for Lease. The Lease can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Lease whose instance is pointed by $obj-Uid

```
Set-BrokerLeaseMetadata -InputObject $obj-Uid -Name "MyMetadataName" -
    Value "1234"
```


EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Lease in the site

```
Get-BrokerLease | Set-BrokerLeaseMetadata -Name "MyMetadataName" -Value
    "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Lease in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerLeaseMetadata 'objname'
```

Parameters

-LeaseId

Specifies the Lease object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Lease objects whose Metadata is to be created/updated.

Type:	Lease[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LeaseName

Specifies the Lease object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerLease

You can pipe the Lease to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerLease

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerLease object.

Related Links

Set-BrokerMachine

March 11, 2024

Sets properties on a machine.

Syntax

```
Set-BrokerMachine
   [-InputObject] <Machine[]>
   [-PassThru]
   [-AssignedClientName <String>]
   [-AssignedIPAddress <String>]
   [-ColorDepth <ColorDepth>]
   [-Description <String>]
   [-HostedMachineId <String>]
   [-HypervisorConnectionUid <Int32>]
   [-IconUid <Int32>]
   [-InMaintenanceMode <Boolean>]
   [-IsReserved <Boolean>]
   [-PublishedName <String>]
   [-SecureIcaRequired <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerMachine
   [-MachineName] <String>
```



```
    [-PassThru]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    [-HostedMachineId <String>]
    [-HypervisorConnectionUid <Int32>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-IsReserved <Boolean>]
    [-PublishedName <String>]
    [-SecureIcaRequired <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerMachine cmdlet sets properties on a machine or set of machines. You can specify a single machine by name or multiple machine instances can be passed to the command by piping or using the -InputObject parameter.

Examples

EXAMPLE 1

This example specifies a machine by name and sets its hypervisor connection.

```
Set-BrokerMachine -MachineName 'MyDomain\MyMachine' -
    HypervisorConnectionUid 3
```

EXAMPLE 2

This example finds all machines in domain MyDomain and sets their hypervisor connections.

```
$machines = Get-BrokerMachine -MachineName 'MyDomain\*'
Set-BrokerMachine -InputObject $machines -HypervisorConnectionUid 3
```

EXAMPLE 3

This example also finds all machines in domain MyDomain and sets their hypervisor connections.

```
Get-BrokerMachine -MachineName 'MyDomain\*' | Set-BrokerMachine -
    HypervisorConnectionUid 3
```


Parameters

-InputObject

The machine instances whose properties you want to set.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

The machine whose properties you want to set.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AssignedClientName

Changes the client name assignment of the machine. Set this to $null to remove the assignment. You can assign machines to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedIPAddress

Changes the client IP address assignment of the machine. Set this to $null to remove the assignment. You can assign machines to multiple users, a single client IP address, or a single client name, but only to one of these categories at a time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

The color depth setting configured on the machine, possible values are:

$null, FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Description of the machine.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HostedMachineId

The unique ID by which the hypervisor recognizes the machine.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorConnectionUid

The hypervisor connection that runs the machine. This may only be set for VMs which are not provisioned by MCS.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

The UID of the machine's icon that is displayed in StoreFront.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Sets whether the machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-IsReserved

Specifies whether the machine should be reserved for special use, for example, for AppDisk preparation. A machine cannot be reserved if it is already a member of a desktop group.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

The name of the machine that is displayed in StoreFront, if the machine has been published. It can be set only for private desktops.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Flag indicating whether SecureICA is required or not when starting a session on the machine.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

You can pipe in the machines whose properties you want to set.

Outputs

None or Citrix.Broker.Admin.SDK.Machine

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Machine object.

Related Links

- about_Broker_Machines
- Get-BrokerMachine
- New-BrokerMachine

Set-BrokerMachineCatalog

March 11, 2024

Moves one or more machines into a different catalog.

Syntax

```
Set-BrokerMachineCatalog
   [-InputObject] <Machine[]>
   [-CatalogUid] <Int32>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerMachineCatalog cmdlet is used to move machines into a different catalog. The following properties of the destination catalog must exactly match those of the machine's current catalog otherwise the command fails:

- AllocationType
- ProvisioningType
- PersistUserChanges
- SessionSupport
- IsRemotePC
- MinimumFunctionalLevel
- PhysicalMachines

Changing a machine's catalog does not change the machine's desktop group membership. There is no effect on user sessions present on a machine if its catalog is changed.

Examples

EXAMPLE 1

This example finds all machines in domain Marketing and moves them into a catalog called MarketingMachines.

```
$catalog = Get-BrokerCatalog MarketingMachines
$machines = Get-BrokerMachine -MachineName 'Marketing\*'
Set-BrokerMachineCatalog $machines -CatalogUid $cat.Uid
```

Parameters

-InputObject

The machine instances that are being moved into a different catalog.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-CatalogUid

The unique identifier of the catalog into which the machines are being moved.

Type:	Int32
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

You can pipe in the machines that are to be moved into a new catalog.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Machines
- Set-BrokerMachine
- New-BrokerCatalog

Set-BrokerMachineCommandMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for MachineCommand

Syntax

```
Set-BrokerMachineCommandMetadata
   [-MachineCommandId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerMachineCommandMetadata
   [-MachineCommandId] <Int64>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineCommandMetadata
    [-MachineCommandId] <Int64>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineCommandMetadata
    [-InputObject] <MachineCommand[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineCommandMetadata
    [-InputObject] <MachineCommand[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerMachineCommandMetadata cmdlet creates/updates metadata key-value pairs for MachineCommand. The MachineCommand can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the MachineCommand whose instance is pointed by $obj-Uid

```
Set-BrokerMachineCommandMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```


EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the MachineCommand in the site

```
1 Get-BrokerMachineCommand | Set-BrokerMachineCommandMetadata -Name "
    MyMetadataName" -Value "1234"
```

Parameters

-MachineCommandId

Specifies the MachineCommand object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-InputObject

Specifies the MachineCommand objects whose Metadata is to be created/updated.

Type:	MachineCommand[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachineCommand

You can pipe the MachineCommand to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerMachineCommand

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerMachineCommand object.

Related Links

Set-BrokerMachineConfiguration

March 11, 2024

Sets the properties of a machine configuration.

Syntax

```
Set-BrokerMachineConfiguration
   [-InputObject] <MachineConfiguration[]>
   [-PassThru]
   [-Description <String>]
   [-Policy <Byte[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerMachineConfiguration
   [-Name] <String>
   [-PassThru]
   [-Description <String>]
   [-Policy <Byte[]>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Sets the properties of a machine configuration. The encoded settings data must only contain settings that match the SettingsGroup of the associated configuration slot. Use the SDK snap-in that matches the SettingsGroup of the associated configuration slot to generate new encoded settings data or modify existing settings values.

Examples

EXAMPLE 1

Use the encoded settings binary data in $newPolicy to update the machine configuration.

```
Set-BrokerMachineConfiguration -Name "UPM\Finance Department" -Policy
   $newPolicy
```


EXAMPLE 2

Assign the description “User Profile Management”to every machine configuration associated with the UPM slot.

```
1  Get-BrokerMachineConfiguration -Name "UPM\*" | Set-
       BrokerMachineConfiguration -Description "User Profile Management"
```

Parameters

-InputObject

Machine configuration to modify.

Type:	MachineConfiguration[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Name of machine configuration to modify.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

New description for the machine configuration.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Policy

New binary array of encoded settings data.

Type:	Byte[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.MachineConfiguration

Machine configuration to modify.

Outputs

None or Citrix.Broker.Admin.SDK.MachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.MachineConfiguration object.

Related Links

- New-BrokerMachineConfiguration
- Get-BrokerMachineConfiguration
- Rename-BrokerMachineConfiguration
- Remove-BrokerMachineConfiguration
- Add-BrokerMachineConfiguration
- about_Broker_ConfigurationSlots

Set-BrokerMachineConfigurationMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for MachineConfiguration

Syntax

```
Set-BrokerMachineConfigurationMetadata
    [-MachineConfigurationId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
    [-MachineConfigurationId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
    [-MachineConfigurationId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
```



```
    [-InputObject] <MachineConfiguration[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
    [-InputObject] <MachineConfiguration[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
    [-MachineConfigurationName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineConfigurationMetadata
    [-MachineConfigurationName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerMachineConfigurationMetadata cmdlet creates/updates metadata key-value pairs for MachineConfiguration. The MachineConfiguration can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the MachineConfiguration whose instance is pointed by $obj-Uid

```
```



```
Set-BrokerMachineConfigurationMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the MachineConfiguration in the site

```
Get-BrokerMachineConfiguration | Set-BrokerMachineConfigurationMetadata
     -Name "MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the MachineConfiguration in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerMachineConfigurationMetadata 'objname'
```

Parameters

-MachineConfigurationId

Specifies the MachineConfiguration object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the MachineConfiguration objects whose Metadata is to be created/updated.

Type:	MachineConfiguration[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineConfigurationName

Specifies the MachineConfiguration object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachineConfiguration

You can pipe the MachineConfiguration to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerMachineConfiguration

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerMachineConfiguration object.

Related Links

Set-BrokerMachineMaintenanceMode

March 11, 2024

Sets whether the specified machine(s) are in maintenance mode.

Syntax

```
1 Set-BrokerMachineMaintenanceMode
2    [-InputObject] <Machine[]>
3    [-MaintenanceMode] <Boolean>
4    [-LoggingId <Guid>]
5    [<CitrixCommonParameters>]
6    [<CommonParameters>]
```


Description

The cmdlet can be used to set whether a machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

There are times when it is necessary to disable desktops. You can do this by setting the InMaintenanceMode property of a desktop to $true. This puts it into maintenance mode. The broker excludes single-session desktops in maintenance mode from brokering decisions and does not start new sessions on them. Existing sessions are unaffected. For multi-session desktops in maintenance mode, reconnections to existing sessions are allowed, but no new sessions are created on the machine.

Desktops in maintenance mode are also excluded from automatic power management, although explicit power actions are still performed.

This cmdlet is equivalent to using the Set-BrokerMachine cmdlet to set the value of only the InMaintenanceMode property.

Examples

EXAMPLE 1

This example finds all machines in domain MyDomain and removes them from maintenance mode by setting their InMaintenanceMode property to false.

```
$machines = Get-BrokerMachine -MachineName 'MyDomain\*'
Set-BrokerMachineMaintenanceMode -InputObject $machines $false
```

Parameters

-InputObject

The machine instances whose InMaintenanceMode property you want to set.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MaintenanceMode

Sets whether the machine is in maintenance mode or not. A machine in maintenance mode is not available for new sessions, and for managed machines all automatic power management is disabled.

Type:	Boolean
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

You can pipe in the machines whose properties you want to set.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Machines
- Set-BrokerMachine

Set-BrokerMachineMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Machine

Syntax

```
Set-BrokerMachineMetadata
   [-MachineId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
   [-MachineId] <Int32>
   [-PassThru]
   -Name <String>
```



```
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
    [-MachineId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
    [-InputObject] <Machine[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
    [-InputObject] <Machine[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
    [-MachineName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerMachineMetadata
    [-MachineName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerMachineMetadata cmdlet creates/updates metadata key-value

pairs for Machine. The Machine can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the Machine whose instance is pointed by $obj-Uid

```
Set-BrokerMachineMetadata -InputObject $obj-Uid -Name "MyMetadataName"
    -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the Machine in the site

```
Get-BrokerMachine | Set-BrokerMachineMetadata -Name "MyMetadataName" -
    Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the Machine in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerMachineMetadata 'objname'
```

Parameters

-MachineId

Specifies the Machine object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Machine objects whose Metadata is to be created/updated.

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Specifies the Machine object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerMachine

You can pipe the Machine to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerMachine

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerMachine object.

Related Links

Set-BrokerPowerTimeScheme

March 11, 2024

Modifies an existing power time scheme.

Syntax

```
Set-BrokerPowerTimeScheme
    [-InputObject] <PowerTimeScheme[]>
    [-PassThru]
    [-DaysOfWeek <TimeSchemeDays>]
    [-DisplayName <String>]
    [-PeakHalfHours <Boolean[]>]
    [-PeakHours <Boolean[]>]
    [-PoolSize <Int32[]>]
    [-PoolSizeHalfHours <Int32[]>]
    [-PoolUsingPercentage <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerPowerTimeScheme
    [-Name] <String>
    [-PassThru]
    [-DaysOfWeek <TimeSchemeDays>]
    [-DisplayName <String>]
    [-PeakHalfHours <Boolean[]>]
    [-PeakHours <Boolean[]>]
    [-PoolSize <Int32[]>]
    [-PoolSizeHalfHours <Int32[]>]
    [-PoolUsingPercentage <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerPowerTimeScheme cmdlet modifies an existing time scheme.

Each power time scheme is associated with a particular desktop group, and covers one or more days of the week, defining which hours of those days are considered peak times and which are off-peak times.In addition, the time scheme defines a pool size value for each hour of the day for the days of the week covered by the time scheme. No one desktop group can be associated with two or more time schemes that cover the same day of the week.

For more information about the power policy mechanism and pool size management, see ‘help about_Broker_PowerManagement’.

Examples

EXAMPLE 1

Sets the pool size for the power time scheme named ‘Development Weekdays’to be 20 for the time between 8am to 6:30pm, and 5 for other times.

```
Set-BrokerPowerTimeScheme -Name 'Development Weekdays' -
    PoolSizeHalfHours ( 0..47 | %{
  if ($_ -lt 16 -or $_ -gt 37) {
  5  }
  else {
  20  }
   }
  )
```

Parameters

-InputObject

The power time scheme to be changed.

Type:	PowerTimeScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Identifies the power time scheme to be changed by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DaysOfWeek

Changes the pattern of days of the week that the time scheme applies to. The pattern of days is specified as a single value or a list of values, where each value refers to either a single day or defined group of days.

Valid values are: Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Weekend and Weekdays.

Type:	TimeSchemeDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisplayName

Changes the name that is used by the DesktopStudio console when showing the time scheme.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakHalfHours

A set of 48 boolean flag values, one for each hour half of the day. The first value in the array relates to midnight to 00:29, the next one to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. If the flag is $true it means that the associated half hour of the day is considered a peak time; if $false it means that it is considered off-peak.

If fewer than 48 values are supplied, the final missing values are assumed to be 'false', and if more than 48 values are supplied, only the first 48 are used.

Type:	Boolean[]
Position:	Named
Default value:	48 $false values, meaning all half hours are off-peak
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PeakHours

Changes the pattern of hours considered 'peak'vs 'off-peak'for the days covered by the time scheme. A 24-entry array of boolean truth values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:59, the first relates to 1am to 1:59 and so on, with the last array element relating to 11 PM to 11:59. If the flag value is $true it means the associated hour of the day is considered a peak time; if $false it means that it is considered off-peak.

If fewer than 24 values are supplied, the final missing values are assumed to be 'false', and if more than 24 values are supplied, only the first 24 are used.

Type:	Boolean[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PoolSize

Changes the requested pool size of running machines at the various hours of the day (for single-session desktop groups, or half hours for multi-session) for the days covered by the time scheme.

For single-session desktop groups, a 24-entry array of integer values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:59, the first relates to 1 AM to 1:59 and so on, with the last array element relating to 11 PM to 11:59. If fewer than 24 values are supplied, the final missing values are assumed to be -1, and if more than 24 values are supplied, only the first 24 are used.

For multi-session desktop groups, a 48-entry array of integer values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:29, the first relates to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. If fewer than 48 values are supplied, the final missing values are assumed to be -1, and if more than 48 values are supplied, only the first 48 are used.

The pool size array entry values are either absolute numbers of machines that should be running or are a percentage of the machines in the desktop group that should be running during the associated hour of the day. A value of -1 in the array signifies that no management of the number of running machines should be attempted during the associated hour of the day.

Type:	Int32[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PoolSizeHalfHours

Changes the requested pool size of running machines at the various half-hours of the day for the days covered by the time scheme.

A 48-entry array of integer values is expected, where the zeroth entry of the array relates to the time period between midnight and 0:29, the first relates to 0:30 AM to 0:59 and so on, with the last array element relating to 11:30 PM to 11:59. If fewer than 48 values are supplied, the final missing values are assumed to be -1, and if more than 48 values are supplied, only the first 48 are used.

The pool size array entry values are either absolute numbers of machines that should be running or are a percentage of the machines in the desktop group that should be running during the associated half hour of the day. A value of -1 in the array signifies that no management of the number of running machines should be attempted during the associated half hour of the day.

Type:	Int32[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PoolUsingPercentage

Changes whether pool size values from the 'PoolSize'or 'PoolSizeHalfHours'array are evaluated as absolute numbers of running machines or as a percentage of machines in the desktop group that are to be maintained as running.

A value of $true indicates that the pool size array values are percentages of total machines in the desktop group and a value of $false indicates that the pool size array values are absolute numbers of machines to maintain as running.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.PowerTimeScheme

The power time scheme to be changed.

Outputs

None or Citrix.Broker.Admin.SDK.PowerTimeScheme

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.PowerTimeScheme object.

Related Links

- about_Broker_PowerManagement
- Get-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme
- New-BrokerPowerTimeScheme
- Remove-BrokerPowerTimeScheme

Set-BrokerPowerTimeSchemeMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for PowerTimeScheme

Syntax

```
Set-BrokerPowerTimeSchemeMetadata
   [-PowerTimeSchemeId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerPowerTimeSchemeMetadata
   [-PowerTimeSchemeId] <Int32>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerPowerTimeSchemeMetadata
   [-PowerTimeSchemeId] <Int32>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerPowerTimeSchemeMetadata
   [-InputObject] <PowerTimeScheme[]>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerPowerTimeSchemeMetadata
   [-InputObject] <PowerTimeScheme[]>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerPowerTimeSchemeMetadata
   [-PowerTimeSchemeName] <String>
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerPowerTimeSchemeMetadata
   [-PowerTimeSchemeName] <String>
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerPowerTimeSchemeMetadata cmdlet creates/updates metadata key-value pairs for PowerTimeScheme. The PowerTimeScheme can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the PowerTimeScheme whose instance is pointed by $obj-Uid

```
Set-BrokerPowerTimeSchemeMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the PowerTimeScheme in the site

```
Get-BrokerPowerTimeScheme | Set-BrokerPowerTimeSchemeMetadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the PowerTimeScheme in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerPowerTimeSchemeMetadata 'objname'
```

Parameters

-PowerTimeSchemeId

Specifies the PowerTimeScheme object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the PowerTimeScheme objects whose Metadata is to be created/updated.

Type:	PowerTimeScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PowerTimeSchemeName

Specifies the PowerTimeScheme object whose Metadata is to be created/updated by name.

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerPowerTimeScheme

You can pipe the PowerTimeScheme to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerPowerTimeScheme

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerPowerTimeScheme object.

Related Links

Set-BrokerPrivateDesktop

March 11, 2024

Change the settings of a private desktop.

Syntax

```
Set-BrokerPrivateDesktop
    [-InputObject] <PrivateDesktop[]>
    [-PassThru]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
```



```
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-PublishedName <String>]
    [-SecureIcaRequired <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerPrivateDesktop
    [-MachineName] <String>
    [-PassThru]
    [-AssignedClientName <String>]
    [-AssignedIPAddress <String>]
    [-ColorDepth <ColorDepth>]
    [-Description <String>]
    [-IconUid <Int32>]
    [-InMaintenanceMode <Boolean>]
    [-PublishedName <String>]
    [-SecureIcaRequired <Boolean>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Private desktops are automatically created when a machine is added to a desktop group with a DesktopKind of 'Private', and these inherit default properties. Use Set-BrokerPrivateDesktop to change the configuration settings of an existing private desktop.

To specify private desktops, you can choose whether to update by machine name, or by passing a PrivateDesktop or an array of PrivateDesktop objects. You can also use the Uid or an array of Uids instead.

You cannot modify many properties of a private desktop as these contain status information; for example DNSName, RegistrationState, and OSVersion.

Use Add- and Remove- cmdlets to update relationships between private desktops and other objects. For example, you can add a tag to a private desktop with:

Add-BrokerTag $tag -Desktop $desktop.Uid

Similarly, assign users to private desktops with:

Add-BrokerUser $user -PrivateDesktop $desktop

Many of the fields that can be set with this cmdlet can also be set with Set-BrokerMachine, such as MaintenanceMode. Using Set-BrokerMachine is preferred in these cases.

For more information about desktops, see about_Broker_Desktops; for more information about machines, see about_Broker_Machines.

Examples

EXAMPLE 1

Change the color depth of Machine1 to be 16-bit.

```
Set-BrokerPrivateDesktop DOMAIN\Machine1 -ColorDepth SixteenBit
```

EXAMPLE 2

Bring all private desktops currently in maintenance mode back into normal service.

```
Get-BrokerPrivateDesktop -InMaintenanceMode $true | Set-
    BrokerPrivateDesktop -InMaintenanceMode $false
```

Parameters

-InputObject

Specifies the desktop or array of desktops to modify. You can also use an integer Uid of the desktop instead.

Type:	PrivateDesktop[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Specifies the desktop to modify using its machine name (in the form 'domain\machine').

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedClientName

Changes the client name assignment of the desktop. Set this to $null to remove the assignment. Desktops can be assigned to multiple users, a single IP address, or a single client name, but only to one of these categories at one time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AssignedIPAddress

Changes the IP address assignment of the desktop. Set this to $null to remove the assignment. Desktops can be assigned to multiple users, a single IP address, or a single client name, but only to one of these categories at one time.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ColorDepth

Changes the color depth connections to this desktop should use.

Valid values are $null, FourBit, EightBit, SixteenBit, and TwentyFourBit. A value of $null results in the desktop group value being used instead.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Changes the description of the desktop.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IconUid

Changes the icon displayed for this desktop. When this setting is $null, the icon displayed is determined by the desktop group.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Changes the maintenance mode setting of a desktop. When a desktop is in maintenance mode, it is not included as a candidate when brokering new sessions, and it does not participate in automatic power management (idle pool); however, it still responds to explicit power operations.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PublishedName

Changes the name displayed to the user for this desktop. When this setting is $null, the name displayed is determined by the PublishedName of the desktop group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Changes whether or not SecureICA is required for connections to this desktop. When this setting is $null, the SecureIcaRequired setting from the desktop group is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.PrivateDesktop

You can pipe PrivateDesktop objects into this cmdlet instead of on the command line with the -InputObject parameter.

Outputs

None or Citrix.Broker.Admin.SDK.PrivateDesktop

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.PrivateDesktop object.

Related Links

- about_Broker_Desktops
- about_Broker_Machines
- Get-BrokerMachine

Set-BrokerRebootCycleMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for RebootCycle

Syntax

```
Set-BrokerRebootCycleMetadata
    [-RebootCycleId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootCycleMetadata
    [-RebootCycleId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootCycleMetadata
    [-RebootCycleId] <Int64>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootCycleMetadata
    [-InputObject] <RebootCycle[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootCycleMetadata
    [-InputObject] <RebootCycle[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerRebootCycleMetadata cmdlet creates/updates metadata key-value pairs for RebootCycle. The RebootCycle can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the RebootCycle whose instance is pointed by $obj-Uid

```
Set-BrokerRebootCycleMetadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the RebootCycle in the site

```
Get-BrokerRebootCycle | Set-BrokerRebootCycleMetadata -Name "
    MyMetadataName" -Value "1234"
```

Parameters

-RebootCycleId

Specifies the RebootCycle object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the RebootCycle objects whose Metadata is to be created/updated.

Type:	RebootCycle[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerRebootCycle

You can pipe the RebootCycle to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerRebootCycle

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerRebootCycle object.

Related Links

Set-BrokerRebootSchedule

March 11, 2024

Updates the values of one or more desktop group reboot schedules.

Syntax

```
Set-BrokerRebootSchedule
   [-InputObject] <RebootSchedule[]>
   [-PassThru]
   [-Day <RebootScheduleDays>]
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   [-RebootDuration <Int32>]
   [-StartTime <TimeSpan>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerRebootSchedule
   [-DesktopGroupName] <String>
   [-PassThru]
   [-Day <RebootScheduleDays>]
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   [-RebootDuration <Int32>]
   [-StartTime <TimeSpan>]
   [-WarningDuration <Int32>]
   [-WarningMessage <String>]
   [-WarningRepeatInterval <Int32>]
   [-WarningTitle <String>]
   [-LoggingId <Guid>]
```



```
14      [<CitrixCommonParameters>]
15      [<CommonParameters>]
```

Description

The Set-BrokerRebootSchedule cmdlet is used to alter the settings of an existing desktop group reboot schedule.

Examples

EXAMPLE 1

Sets the reboot schedule for the desktop group named Accounting to display a message with the title "WARNING: Reboot pending"and body "Save your work"ten minutes prior to rebooting each machine. The message is displayed in every user session on that machine.

```
1  Set-BrokerRebootSchedule -DesktopGroupName Accounting -WarningMessage "
       Save your work" -WarningDuration 10 -WarningTitle "WARNING: Reboot
       pending"
```

EXAMPLE 2

Sets all weekly reboot schedules to run on Friday.

```
1  Get-BrokerRebootSchedule -Frequency Weekly | Set-BrokerRebootSchedule -
       Day Friday
```

EXAMPLE 3

Sets the reboot schedule for the desktop group having Uid 17 to display the message "Rebooting in %m% minutes."Fifteen, ten and five minutes prior to rebooting each machine, the message "Rebooting in %m% minutes."will be displayed in each user session with the pattern '%m%'replaced with the number of minutes until the reboot.

```
1  Set-BrokerRebootSchedule -DesktopGroupUid 17 -WarningMessage "Rebooting
       in %m% minutes." -WarningDuration 15 -WarningRepeatInterval 5
```


Parameters

-InputObject

The reboot schedule to be modified.

Type:	RebootSchedule[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose reboot schedule is to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Day

For weekly schedules, the day of the week on which the scheduled reboot-cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the reboot schedule is to be enabled or disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Frequency

Frequency with which this schedule runs (either Weekly or Daily).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootSchedule

Reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerRebootSchedule
- New-BrokerRebootSchedule
- Remove-BrokerRebootSchedule

Set-BrokerRebootScheduleV2

March 11, 2024

Updates the values of one or more desktop group reboot schedules.

Syntax

```
Set-BrokerRebootScheduleV2
   [-InputObject] <RebootScheduleV2[]>
   [-PassThru]
   [-Day <RebootScheduleDays>]
   [-DayInMonth <RebootScheduleDays>]
   [-Description <String>]
   [-Enabled <Boolean>]
   [-Frequency <RebootScheduleFrequency>]
   [-FrequencyFactor <Int32>]
   [-IgnoreMaintenanceMode <Boolean>]
   [-MaxOvertimeStartMins <Int32>]
   [-RebootDuration <Int32>]
   [-RestrictToTag <String>]
   [-StartDate <String>]
   [-StartTime <TimeSpan>]
   [-UseNaturalReboot <Boolean>]
```



```
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-WeekInMonth <RebootScheduleWeeks>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2
    [-Name] <String>
    [-PassThru]
    [-Day <RebootScheduleDays>]
    [-DayInMonth <RebootScheduleDays>]
    [-Description <String>]
    [-Enabled <Boolean>]
    [-Frequency <RebootScheduleFrequency>]
    [-FrequencyFactor <Int32>]
    [-IgnoreMaintenanceMode <Boolean>]
    [-MaxOvertimeStartMins <Int32>]
    [-RebootDuration <Int32>]
    [-RestrictToTag <String>]
    [-StartDate <String>]
    [-StartTime <TimeSpan>]
    [-UseNaturalReboot <Boolean>]
    [-WarningDuration <Int32>]
    [-WarningMessage <String>]
    [-WarningRepeatInterval <Int32>]
    [-WarningTitle <String>]
    [-WeekInMonth <RebootScheduleWeeks>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerRebootScheduleV2 cmdlet is used to alter the settings of an existing desktop group reboot schedule.

Examples

EXAMPLE 1

Sets the reboot schedule for the reboot schedule named Accounting to display a message with the title "WARNING: Reboot pending"and body "Save your work"ten minutes prior to rebooting each machine. The message is displayed in every user session on that machine.


```
Set-BrokerRebootScheduleV2 -Name Accounting -WarningMessage "Save your
    work" -WarningDuration 10 -WarningTitle "WARNING: Reboot pending"
```

EXAMPLE 2

Sets all weekly reboot schedules to run on Friday.

```
Get-BrokerRebootScheduleV2 -Frequency Weekly | Set-
    BrokerRebootScheduleV2 -Day Friday
```

EXAMPLE 3

Sets the reboot schedule for the reboot schedule having Uid 17 to display the message "Rebooting in %m% minutes."Fifteen, ten and five minutes prior to rebooting each machine, the message "Rebooting in %m% minutes."will be displayed in each user session with the pattern '%m%'replaced with the number of minutes until the reboot.

```
Set-BrokerRebootScheduleV2 17 -WarningMessage "Rebooting in %m% minutes
    ." -WarningDuration 15 -WarningRepeatInterval 5
```

EXAMPLE 4

Restricts the reboot schedule having uid 12 to only reboot machines within the desktop group tagged with the 'Finance Dept'tag.

```
Set-BrokerRebootScheduleV2 12 -RestrictToTag 'Finance Dept'
```

Parameters

-InputObject

The reboot schedule to be modified.

Type:	RebootScheduleV2[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

The name of the reboot schedule

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Day

For weekly schedules, the days of the week on which the scheduled reboot-cycle starts (one or more of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DayInMonth

For monthly schedules, the day in the month on which the scheduled reboot-cycle starts (one of Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday).

Type:	RebootScheduleDays
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

An optional description for the reboot schedule.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the reboot schedule is to be enabled or disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Frequency

Frequency with which this schedule runs (either Weekly or Daily or Monthly).

Type:	RebootScheduleFrequency
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FrequencyFactor

The frequency factor for the reboot schedule. It indicates how often the schedule should run with respect to the frequency. For example, if the FrequencyFactor is set to 2, it means every two days from the StartDate when the Frequency is Daily, every two weeks from the StartDate when the Frequency is Weekly, and similarly for monthly. It defaults to 1 if not provided. The StartDate needs to be set if the frequency factor is greater than 1.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreMaintenanceMode

Boolean value to reboot machines in maintenance mode

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxOvertimeStartMins

Maximum delay in minutes after which the scheduled reboot will not take place

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RebootDuration

Approximate maximum number of minutes over which the scheduled reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTag

If specified, restricts the reboot schedule to only those machines in the desktop group with the specified tag. Specify $null to remove any tag restriction.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDate

The date on which the first schedule is expected to run, date is in ISO 8601 Format (YYYY-MM-DD). The actual start date to match the frequency pattern need not be provided. For example, if today is Monday and the schedule is set to run every Sunday for every four weeks, there is no need to identify the date on which Sunday falls after four weeks. Today's date ((Get-Date).Date.ToString('yyyy-MM-dd')) can be provided and the service would adjust the start date to reflect the actual date i.e, the Sunday four weeks from today. Get-BrokerRebootScheduleV2 cmdlet will reflect the actual start date.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Time of day at which the scheduled reboot cycle starts (HH:MM).

Type:	TimeSpan
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseNaturalReboot

Boolean value indicating whether the machines should reboot whenever they happen to have no sessions, rather than at equally spaced times within the cycle duration.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningDuration

Time prior to the initiation of a machine reboot at which warning message is displayed in all user sessions on that machine. If the warning duration is zero then no message is displayed. In some cases the time required to process a reboot schedule may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Time to wait after the previous reboot warning before displaying the warning message in all user sessions on that machine again. If the warning repeat interval is zero then the warning message is not displayed after the initial warning.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WeekInMonth

For monthly schedules, the week in the month on which the scheduled reboot-cycle starts (one of First, Second, Third, Fourth, Last).

Type:	RebootScheduleWeeks
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootScheduleV2

Reboot schedules may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerRebootScheduleV2
- New-BrokerRebootScheduleV2
- Remove-BrokerRebootScheduleV2
- Rename-BrokerRebootScheduleV2

Set-BrokerRebootScheduleV2Metadata

March 11, 2024

Creates/Updates metadata key-value pairs for RebootScheduleV2

Syntax

```
Set-BrokerRebootScheduleV2Metadata
    [-RebootScheduleV2Id] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-RebootScheduleV2Id] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-RebootScheduleV2Id] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-InputObject] <RebootScheduleV2[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-InputObject] <RebootScheduleV2[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-RebootScheduleV2Name] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerRebootScheduleV2Metadata
    [-RebootScheduleV2Name] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerRebootScheduleV2Metadata cmdlet creates/updates metadata key-value pairs for RebootScheduleV2. The RebootScheduleV2 can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata "MyMetadataName"key-value pair for the RebootScheduleV2 whose instance is pointed by $obj-Uid

```
Set-BrokerRebootScheduleV2Metadata -InputObject $obj-Uid -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key "MyMetadataName"with the value "1234"for all the RebootScheduleV2 in the site

```
Get-BrokerRebootScheduleV2 | Set-BrokerRebootScheduleV2Metadata -Name "
    MyMetadataName" -Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys "name1"and "name2"with the values "value1" and "value2"respectively for the RebootScheduleV2 in the site whose name is 'objname'

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerRebootScheduleV2Metadata 'objname'
```


Parameters

-RebootScheduleV2Id

Specifies the RebootScheduleV2 object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the RebootScheduleV2 objects whose Metadata is to be created/updated.

Type:	RebootScheduleV2[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-RebootScheduleV2Name

Specifies the RebootScheduleV2 object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerRebootScheduleV2

You can pipe the RebootScheduleV2 to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerRebootScheduleV2

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerRebootScheduleV2 object.

Related Links

Set-BrokerRemotePCAccount

March 11, 2024

Modify one or more RemotePCAccounts.

Syntax

```
Set-BrokerRemotePCAccount
   [-InputObject] <RemotePCAccount[]>
   [-PassThru]
   [-AllowSubfolderMatches <Boolean>]
   [-MachinesExcluded <String[]>]
   [-MachinesIncluded <String[]>]
   [-OU <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Modify one or more RemotePCAccounts.

Examples

EXAMPLE 1

Make all RemotePCAccounts filter out machines from DOMAIN42.

```
Get-BrokerRemotePCAccount | Set-BrokerRemotePCAccount -MachinesExcluded
    @('DOMAIN42\*')
```

Parameters

-InputObject

Specifies the RemotePCAccounts to modify.

Type:	RemotePCAccount[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AllowSubfolderMatches

When true a machine matches this RemotePCAccount if the AD computer is in the container specified by the OU property, or within a child container of the OU.

When false the AD computer object only matches if it is directly in the AD container specified by the OU property.

This property is not meaningful when OU has the special value 'any'.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MachinesExcluded

MachinesExcluded specifies a set of strings that can include asterisk wildcards. If a machine name matches any entries in MachinesExcluded then it cannot match with this RemotePCAccount regardless of whether there is a MachinesIncluded match.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

DOMAIN1\M*

DOMAIN*\M*

\M

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesIncluded

MachinesIncluded specifies a set of strings that can include asterisk wildcards. A machine may only match with this RemotePCAccount if it matches a MachinesIncluded entry and does not match any MachinesExcluded entries.

Matches are performed against the domain name joined with the machine name by a backslash (DOMAIN\MACHINE), e.g.:

DOMAIN1\M*

DOMAIN*\M*

\M

Type:	String[]

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OU

Specifies the DN of an AD container, or has the special value 'any'.

When an AD container is specified a machine may only match with the RemotePCAccount when the AD computer object is located relative to the OU.

When 'any'is specified the location of the AD computer object is ignored for purposes of matching this RemotePCAccount. The machine must still meet the MachinesIncluded and MachinesExcluded filters for a match to occur.

In the event that a machine matches with multiple RemotePCAccounts then the RemotePCAccount OU with the longest canonical name takes precedence. The special 'any'OU is treated as lowest priority.

Note that the OU value of every RemotePCAccount must be unique, and this includes only one 'any' entry being permitted.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RemotePCAccount

You can pipe the RemotePCAccounts to be modified into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.RemotePCAccount

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.RemotePCAccount object.

Related Links

- about_Broker_RemotePC
- Get-BrokerRemotePCAccount

- New-BrokerRemotePCAccount
- Remove-BrokerRemotePCAccount

Set-BrokerServiceConfigurationData

March 11, 2024

Set the Service configuration data objects.

Syntax

```
Set-BrokerServiceConfigurationData
    [-InputObject] <ServiceConfigurationData[]>
    [-PassThru]
    [-SettingValue <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

View the Service configuration data objects stored in the broker database. These objects if set correctly, if set overrride the values present in the windows registry

Examples

EXAMPLE 1

Running this will set the HeartbeatPeriodMs to 6 minutes

```
Set-BrokerServiceConfigurationData Core.HeartbeatPeriodMs  -
    SettingValue 360000
```

EXAMPLE 2

Running this will reset the value of MaxHeartbeatIntervalMs and it will be read from the windows registry instead of the Broker Database

```
Set-BrokerServiceConfigurationData MaxHeartbeatIntervalMs
```


Parameters

-InputObject

The service configuration data object

Type:	ServiceConfigurationData[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SettingValue

The value of the service setting that can be overridden and stored in the database. This must fall in the specified range as described in order to successfully set the service configuration object. Keeping the parameter as null will result in resetting of the key(It will be read from the windows registry in that case)

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

Input cannot be piped to this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.ServiceConfigurationData

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.ServiceConfigurationData object.

Related Links

- Get-BrokerServiceConfigurationData

Set-BrokerSession

March 11, 2024

Sets properties of a session.

Syntax

```
Set-BrokerSession
   [-InputObject] <Session[]>
   [-PassThru]
   [-Hidden <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSession
   [-SessionKey] <Guid>
   [-PassThru]
   [-Hidden <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerSession cmdlet sets properties on a session or set of sessions. You can specify a single session by Uid or SessionKey (Guid) or multiple session instances can be passed to the command by piping or using the -InputObject parameter.

Examples

EXAMPLE 1

Finds all sessions in the site for user John Smith that have been hidden, and makes them available for reconection again.

```
$sessions = Get-BrokerSession -User ACCOUNTS\JohnSmith -Hidden $true
$sessions | Set-BrokerSession -Hidden $false
```

Parameters

-InputObject

The session instances whose properties you want to set.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionKey

The session key (Guid) of the session whose properties you want to set.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Hidden

Changes whether the session is hidden or not. Hidden sessions are treated as though they do not exist when brokering sessions; a hidden session cannot be reconnected to, but a new session may be launched using the same entitlement.

A session may be hidden automatically by the system when an attempt is made to reconnect to a session that is unreachable due to, for example, the failure of a VM's hypervisor. This allows a new session to be brokered provided that other machines in the same desktop group are still available. If the original session still exists after the hypervisor is restored then it must be unhidden to allow the user to reconnect to it.

Only sessions for shared desktops or applications can be hidden or unhidden. Private desktop or application sessions can never be hidden.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

You can pipe in the sessions whose properties you want to set.

Outputs

None or Citrix.Broker.Admin.SDK.Session

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Session object.

Related Links

- Get-BrokerSession

Set-BrokerSessionLinger

March 11, 2024

Updates the values of one or more desktop group session linger settings.

Syntax

```
Set-BrokerSessionLinger
   [-InputObject] <SessionLinger[]>
   [-PassThru]
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSessionLinger
   [-DesktopGroupName] <String>
   [-PassThru]
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerSessionLinger cmdlet is used to alter the settings of an existing desktop group session linger setting.

Examples

EXAMPLE 1

Sets the disconnect time for the session linger setting associated with desktop group named Accounting.

```
Set-BrokerSessionLinger -DesktopGroupName Accounting -
    MaxTimeBeforeDisconnect 0:10
```

Parameters

-InputObject

The session linger to be modified.

Type:	SessionLinger[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose session linger setting is to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the session linger setting is to be enabled or disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Specifies the average load threshold across the desktop group. When the threshold hits, lingering sessions across the group be terminated to reduce load. Sessions that have been lingering the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Specifies the maximum load threshold per machine in the desktop group. When the threshold hits, lingering sessions on each loaded machine will be terminated to reduce load. Sessions that have been lingering the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Specifies the time by which a lingering session will be disconnected. The disconnect time cannot be greater than the terminate timer (if enabled). When the disconnect and terminate times are the same, the terminate time takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Specifies the time by which a lingering session will be terminated. When the disconnect and terminate times are the same, the terminate time takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Specifies whether the session linger's user filter is enabled or disabled. Where the user filter is enabled, lingering is enabled only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.SessionLinger

Session linger settings may be specified through pipeline input.

Outputs

None or Citrix.Broker.Admin.SDK.SessionLinger

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.SessionLinger object.

Related Links

- New-BrokerSessionLinger
- Get-BrokerSessionLinger
- Remove-BrokerSessionLinger

Set-BrokerSessionMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Session

Syntax

```
Set-BrokerSessionMetadata
    [-SessionId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-SessionId] <Int64>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-SessionId] <Int64>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-InputObject] <Session[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-InputObject] <Session[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-SessionName] <Guid>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerSessionMetadata
    [-SessionName] <Guid>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerSessionMetadata cmdlet creates/updates metadata key-value

pairs for Session. The Session can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Session whose instance is pointed by $obj-Uid

```
Set-BrokerSessionMetadata -InputObject $obj-Uid -Name "MyMetadataName"
    -Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Session in the site

```
Get-BrokerSession | Set-BrokerSessionMetadata -Name "MyMetadataName" -
    Value "1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Session in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerSessionMetadata 'objname'
```


Parameters

-SessionId

Specifies the Session object whose Metadata is to be created/updated by ID.

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int64
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Session objects whose Metadata is to be created/updated.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SessionName

Specifies the Session object whose Metadata is to be created/updated by name.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerSession

You can pipe the Session to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerSession

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerSession object.

Related Links

Set-BrokerSessionPreLaunch

March 11, 2024

Updates the values of one or more desktop group session pre-launch settings.

Syntax

```
Set-BrokerSessionPreLaunch
   [-InputObject] <SessionPreLaunch[]>
   [-PassThru]
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSessionPreLaunch
   [-DesktopGroupName] <String>
   [-PassThru]
   [-Enabled <Boolean>]
   [-MaxAverageLoadThreshold <Int32>]
   [-MaxLoadPerMachineThreshold <Int32>]
   [-MaxTimeBeforeDisconnect <TimeSpan>]
   [-MaxTimeBeforeTerminate <TimeSpan>]
   [-UserFilterEnabled <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerSessionPreLaunch cmdlet is used to alter the settings of an existing desktop group session pre-launch setting.

Examples

EXAMPLE 1

Sets the disconnect time for the session pre-launch setting associated with desktop group named Accounting.

```
Set-BrokerSessionPreLaunch -DesktopGroupName Accounting -
    MaxTimeBeforeDisconnect 0:10
```

Parameters

-InputObject

The session pre-launch to be modified.

Type:	SessionPreLaunch[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DesktopGroupName

The name of the desktop group whose session pre-launch setting is to be modified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean that indicates if the session pre-launch setting is to be enabled or disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAverageLoadThreshold

Specifies the average load threshold across the desktop group. When the threshold hits, pre-launched sessions across the group be terminated to reduce load. Sessions that have been pre-launched the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxLoadPerMachineThreshold

Specifies the maximum load threshold per machine in the desktop group. When the threshold hits, pre-launched sessions on each loaded machine will be terminated to reduce load. Sessions that have been pre-launched the longest will be chosen first.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeDisconnect

Specifies the time by which a pre-launched session will be disconnected. The disconnect time cannot be greater than the terminate timer (if enabled). When the disconnect and terminate times are the same, the terminate time takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxTimeBeforeTerminate

Specifies the time by which a pre-launched session will be terminated. When the disconnect and terminate times are the same, the terminate time takes precedence.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserFilterEnabled

Specifies whether the session pre-launch's user filter is enabled or disabled. Where the user filter is enabled, pre-launch is enabled only to users who appear in the filter (either explicitly or by virtue of group membership).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.SessionPreLaunch

Session pre-launch settings may be specified through pipeline input.

Outputs

None or Citrix.Broker.Admin.SDK.SessionPreLaunch

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.SessionPreLaunch object.

Related Links

- New-BrokerSessionPreLaunch
- Get-BrokerSessionPreLaunch
- Remove-BrokerSessionPreLaunch

Set-BrokerSharedDesktop

March 11, 2024

Change the settings of a shared desktop.

Syntax

```
Set-BrokerSharedDesktop
   [-InputObject] <SharedDesktop[]>
   [-PassThru]
   [-InMaintenanceMode <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSharedDesktop
   [-MachineName] <String>
   [-PassThru]
   [-InMaintenanceMode <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Shared desktops are automatically created when a machine is added to a desktop group with a DesktopKind of 'Shared', and these inherit default properties. Use Set-BrokerSharedDesktop to change the configuration settings of an existing shared desktop.

To specify shared desktops, you can choose whether to update by machine name or by passing a SharedDesktop or an array of SharedDesktop objects. You can also use the Uid or an array of Uids instead.

Most properties of a shared desktop cannot be modified as these contain status information; for example DNSName, RegistrationState, and OSVersion. You can change only the maintenance mode setting with this cmdlet.

Many of the properties that can be set with Set-BrokerSharedDesktop can be set by using %5BSet-BrokerMachine%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/Broker/Set-BrokerMachine.html) (e.g. InMaintenanceMode). Using the %5BSet-BrokerMachine%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/Broker/Set-BrokerMachine.html) cmdlet, where possible, is the preferred behaviour.

See about_Broker_Desktops for more information about desktops.

Examples

EXAMPLE 1

Put the Machine1 shared desktop into maintenance mode.


```
Set-BrokerSharedDesktop DOMAIN\Machine1 -InMaintenanceMode $true
```

EXAMPLE 2

Bring all shared desktops currently in maintenance mode back into normal service.

```
Get-BrokerSharedDesktop -InMaintenanceMode $true | Set-
    BrokerSharedDesktop -InMaintenanceMode $false
```

Parameters

-InputObject

Specifies the desktop or array of desktops to modify. You can also use an integer Uid of the desktop instead.

Type:	SharedDesktop[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MachineName

Specifies the desktop to modify using its machine name (in the form 'domain\machine').

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InMaintenanceMode

Changes the maintenance mode setting of a desktop. When a desktop is in maintenance mode, it is not included as a candidate when brokering new sessions, and it does not participate in automatic power management (idle pool); however, it still responds to explicit power operations.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.SharedDesktop

You can pipe SharedDesktop objects into this cmdlet instead of on the command line with the -InputObject parameter.

Outputs

None or Citrix.Broker.Admin.SDK.SharedDesktop

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.SharedDesktop object.

Related Links

- about_Broker_Desktops
- Get-BrokerSharedDesktop

Set-BrokerSite

March 11, 2024

Changes the overall settings of the current XenDesktop broker site.

Syntax

```
Set-BrokerSite
   [-PassThru]
   [-AlwaysBypassAuthForCachedResources <Boolean>]
   [-ApplicationIconUid <Int32>]
   [-BaseOU <Guid>]
   [-BypassAuthForCachedResources <Boolean>]
   [-CloudSiteLicense <String>]
   [-CloudValidLicenses <String>]
   [-ColorDepth <ColorDepth>]
   [-ConnectionLeasingEnabled <Boolean>]
   [-CredentialForwardingToCloudAllowed <Boolean>]
   [-DefaultMinimumFunctionalLevel <FunctionalLevel>]
   [-DefaultReuseMachinesWithoutShutdownInOutage <Boolean>]
   [-DeleteResourceLeasesOnLogOff <Boolean>]
   [-DesktopGroupIconUid <Int32>]
   [-DnsResolutionEnabled <Boolean>]
   [-LocalHostCacheEnabled <Boolean>]
   [-PreferredAccountName <String>]
   [-RequireXmlServiceKeyForNFuse <Boolean>]
   [-RequireXmlServiceKeyForSta <Boolean>]
   [-ResourceLeaseValidityPeriodInDays <Int32>]
   [-ResourceLeasingEnabled <Boolean>]
   [-ReuseMachinesWithoutShutdownInOutageAllowed <Boolean>]
   [-SecureIcaRequired <Boolean>]
   [-TelemetryHeadlessLaunchEnabled <Boolean>]
   [-TelemetryLaunchMinTimeIntervalMins <Int32>]
   [-TelemetryLaunchShadowDelayMins <Int32>]
   [-TrustManagedAnonymousXmlServiceRequests <Boolean>]
   [-TrustRequestsSentToTheXmlServicePort <Boolean>]
   [-UseVerticalScalingForRdsLaunches <Boolean>]
   [-XmlServiceKey1 <String>]
   [-XmlServiceKey2 <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerSite cmdlet modifies properties of the current broker site.

The broker site is a top-level, logical representation of the XenDesktop site, from the perspective of the brokering services running within the site. It defines various site-wide default attributes used by the brokering services.

A XenDesktop installation has only a single broker site instance.

Examples

EXAMPLE 1

Specifies that any new desktop groups created, where a color depth value is not specified, default to using 16-bit color depth for user sessions to desktops or applications within that group.

```
1  Set-BrokerSite -ColorDepth SixteenBit
```

Parameters

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlwaysBypassAuthForCachedResources

Allows client to always display cached resources without authentication..

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ApplicationIconUid

Changes the default icon used for new applications if no icon is specified explicitly when an application is created. Changing this default has no impact on the icons used by existing applications.

The specified icon must already have been added to the site using New-BrokerIcon.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BaseOU

Changes the objectGUID property identifying the base OU in Active Directory used for desktop registrations. For sites using only registry-based discovery (the default) this value is $null.

Any desktop attempting to register through a different OU from the one specified here is rejected. Note that desktops configured for registry-based discovery can register with the site, even if a BaseOU value is specified.

Information held in Active Directory is not modified by changing this value.

Typically, this property is changed only by using the Set-ADControllerDiscovery.ps1 script.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BypassAuthForCachedResources

Allows client to display cached resources without authentication..

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CloudSiteLicense

Used to override the Product, Edition and LicensingModel set during provisioning at CCS level.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CloudValidLicenses

List of license SKUs purchased by the customer needs to be set a provisioning time every time Ptah re-provisions the customer because of SKU changes for the customer. This will be used for validation purposes.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ColorDepth

Changes the default color depth for new desktop groups, if no color depth is specified explicitly when a group is created. Changing this default has no impact on the color depths used already by existing groups.

Valid values are FourBit, EightBit, SixteenBit, and TwentyFourBit.

Type:	ColorDepth
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConnectionLeasingEnabled

Connection leasing is no longer supported and cannot be enabled. This property exists for backwards compatibility only.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CredentialForwardingToCloudAllowed

Determines whether to allow the Connector to forward user credentials to cloud for verification when they cannot be authenticated locally.

With the default value ($false), requests from Storefront containing user credentials that cannot be authenticated locally will be failed.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DefaultMinimumFunctionalLevel

Changes the default minimum functional level used for new catalogs and desktop groups when no explicit value is provided.

Type:	FunctionalLevel
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DefaultReuseMachinesWithoutShutdownInOutage

The default ReuseMachinesWithoutShutdownInOutage used for new desktop groups when no explicit value is provided.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeleteResourceLeasesOnLogOff

Enables lease syncing on client.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DesktopGroupIconUid

Changes the default desktop icon used for new desktop groups if no icon is specified explicitly when a group is created. Changing this default has no impact on the icons used already by existing groups.

The specified icon must already have been added to the site using New-BrokerIcon.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DnsResolutionEnabled

Changes whether ICA files returned by a broker service to a user device contain the numeric IP address or the DNS name of the desktop machine to which a session should be established.

With the default value ($false), ICA files will always contain a numeric IP address. To have DNS names appear in the ICA files, set the value to $true.

Even when DNS resolution is enabled ($true), IP addresses may still appear in ICA files. The reasons for this include, for example, that the broker service is unable to obtain a DNS name for the target machine, or that Storefront is configured to always use numeric IP addresses in this context.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalHostCacheEnabled

If the Local Host Cache feature is available, this property enables or disables it at run-time.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreferredAccountName

Determines if SAM name or UPN should be displayed for default name of user/group account

Type:	String
Accepted values:	SamName, SamNameFallbackToUpn, Upn
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RequireXmlServiceKeyForNFuse

Determines whether an XML Service Key header is required for the NFuse, MCP, and Admin XML interfaces.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RequireXmlServiceKeyForSta

Determines whether an XML Service Key header is required for the STA interface.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceLeaseValidityPeriodInDays

Validity period for a lease.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceLeasingEnabled

Enables lease syncing on client.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReuseMachinesWithoutShutdownInOutageAllowed

Allows the ReuseMachinesWithoutShutdownInOutage setting on individual DesktopGroups to be enabled. Because these settings have potential security implications, only the site administrator can enable use of this feature. Disabling this setting will clear corresponding field on all delivery groups.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecureIcaRequired

Changes the default SecureICA usage requirements for new desktop groups if no SecureICA setting is specified explicitly when a group is created. Changing this default has no impact on the SecureICA usage requirements of existing groups.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-TelemetryHeadlessLaunchEnabled

Enables client to perform headless telemetry launches.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TelemetryLaunchMinTimeIntervalMins

Configures minimum time interval (in minutes) between headless telemetry launches.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TelemetryLaunchShadowDelayMins

Configures delay (in minutes) between ICA-HDX launch and headless telemetry launch.

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TrustManagedAnonymousXmlServiceRequests

Changes whether the XML Service (as used by Storefront) implicitly trusts managed anonymous launch requests.

With the default value ($false), any attempt to use the XML service for managed anonymous sessions is rejected.

If this setting is enabled, anyone with access to the XML service will be able to utilize the managed anonymous functionality to leave disconnected prelaunched anonymous sessions available for reconnection. You must ensure that controllers running the brokering services are securely firewalled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TrustRequestsSentToTheXmlServicePort

Changes whether the XML Service (as used by Storefront) implicitly trusts the originator of requests it receives, or whether it fully authenticates them.

With the default value ($false), full authentication checks are performed. However, you must enable this setting ($true) to allow support for “Pass-through”authentication, and/or connections routed through Access Gateway.

If this setting is enabled, you must ensure that controllers running the brokering services are securely firewalled.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseVerticalScalingForRdsLaunches

Determines whether to use vertical scaling when considering RDS machines for launches. Vertical scaling would saturate machines in the current pool rather than send sessions to the least loaded machines. This would be a trade in performance vs. cost, where vertical scaling would be less costly.

With the default value ($false), horizontal scaling will be used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-XmlServiceKey1

A 256 bit service key for the XML Services.

Use the New-BrokerXmlServiceKey cmdlet to generate keys.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-XmlServiceKey2

A 256 bit service key for the XML Services.

Use the New-BrokerXmlServiceKey cmdlet to generate keys.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None or Citrix.Broker.Admin.SDK.Site

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Site object.

Related Links

- about_Broker_Concepts
- Get-BrokerSite
- New-BrokerIcon

Set-BrokerSiteMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Site

Syntax

```
Set-BrokerSiteMetadata
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-BrokerSiteMetadata
   [[-InputObject] <Site[]>]
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSiteMetadata
   [[-InputObject] <Site[]>]
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSiteMetadata
   [-PassThru]
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerSiteMetadata
   [-PassThru]
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerSiteMetadata cmdlet creates/updates metadata key-value

pairs for Site. The Site can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Site

```
Set-BrokerSiteMetadata -Name "MyMetadataName" -Value "1234"
```


EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”

```
Get-BrokerSite | Set-BrokerSiteMetadata -Name "MyMetadataName" -Value "
    1234"
```

Parameters

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Site objects whose Metadata is to be created/updated.

Type:	Site[]
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerSite

You can pipe the Site to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerSite

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerSite object.

Related Links

Set-BrokerTag

March 11, 2024

Sets the properties of a tag.

Syntax

```
Set-BrokerTag
    [-InputObject] <Tag[]>
    [-PassThru]
    [-Description <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTag
    [-Name] <String>
    [-PassThru]
    [-Description <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerTag cmdlet sets properties of a tag or set of tags. The tags on which to operate can be specified by name, in which case multiple tags may be specified through the use of wildcards, or by passing one or more Tag instances to the cmdlet by piping or by using the -InputObject parameter.

Examples

EXAMPLE 1

This example sets the description for the existing tag MyTag.

```
$mytag = Get-BrokerTag MyTag
Set-BrokerTag -InputObject $mytag -Description "This is my tag. I use
    it to label all my things."
```


Parameters

-InputObject

Specifies the tag objects to modify.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Identifies the tag to modify.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Description

Supplies the new value of the Description property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Tag

You can pipe the tags to be modified to Set-BrokerTag.

Outputs

None or Citrix.Broker.Admin.SDK.Tag

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.Tag object.

Notes

A tag's Name property cannot be changed by Set-BrokerTag. To rename a tag use Rename-BrokerTag.

Related Links

- Add-BrokerTag
- Get-BrokerTag
- New-BrokerTag
- Remove-BrokerTag
- Rename-BrokerTag

Set-BrokerTagMetadata

March 11, 2024

Creates/Updates metadata key-value pairs for Tag

Syntax

```
Set-BrokerTagMetadata
    [-TagId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-TagId] <Int32>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-TagId] <Int32>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-InputObject] <Tag[]>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-InputObject] <Tag[]>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-TagName] <String>
    [-PassThru]
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-BrokerTagMetadata
    [-TagName] <String>
    [-PassThru]
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-BrokerTagMetadata cmdlet creates/updates metadata key-value

pairs for Tag. The Tag can be specified by InputObject or piping.

Examples

EXAMPLE 1

This command creates/updates the Metadata “MyMetadataName”key-value pair for the Tag whose instance is pointed by $obj-Uid

```
Set-BrokerTagMetadata -InputObject $obj-Uid -Name "MyMetadataName" -
    Value "1234"
```

EXAMPLE 2

This command creates/updates metadata key “MyMetadataName”with the value “1234”for all the Tag in the site

```
Get-BrokerTag | Set-BrokerTagMetadata -Name "MyMetadataName" -Value "
    1234"
```

EXAMPLE 3

This command creates/updates two metadata keys “name1”and “name2”with the values “value1” and “value2”respectively for the Tag in the site whose name is ‘objname’

```
@{
  'name1' = 'value1'; 'name2' = 'value2'  }
  | Set-BrokerTagMetadata 'objname'
```


Parameters

-TagId

Specifies the Tag object whose Metadata is to be created/updated by ID.

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InputObject

Specifies the Tag objects whose Metadata is to be created/updated.

Type:	Tag[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-TagName

Specifies the Tag object whose Metadata is to be created/updated by name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Name

Specifies the name of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Value

Specifies the value of the Metadata member to be created/updated

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Map

Specifies a hashtable containing name/value pairs to be used to create or update Metadata members

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.BrokerTag

You can pipe the Tag to hold the new or updated metadata.

Outputs

None or Citrix.Broker.Admin.SDK.BrokerTag

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.BrokerTag object.

Related Links

Set-BrokerUserZonePreference

March 11, 2024

Changes the zone preference associated with a user/group account in this site

Syntax

```
Set-BrokerUserZonePreference
   [-InputObject] <UserZonePreference[]>
   [-PassThru]
   [-HomeZoneUid <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-BrokerUserZonePreference
   [-Name] <String>
   [-PassThru]
   [-HomeZoneUid <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-BrokerUserZonePreference cmdlet allows the preferred home zone for resources launched using the specified user/group account to be changed.

Subject to the configuration of the desktop groups in use, and the availability of machines in the preferred zone, desktops and applications are launched using machines in that zone where possible.

Examples

EXAMPLE 1

Changes the preferred zone for resources launched by members of the APAC\marketing group account.

```
Set-BrokerUserZonePreference APAC\marketing -HomeZoneUid 2E885C02-6B65
   -47AA-8B03-E855BE2FF7D7
```


Parameters

-InputObject

The account zone preference to be modified.

Type:	UserZonePreference[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The name of the user/group account whose home zone preference is to be changed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the affected record.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-HomeZoneUid

The home zone preference to be associated with the user/group account for this site.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.UserZonePreference

The account zone preference to be modified.

Outputs

None or Citrix.Broker.Admin.SDK.UserZonePreference

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Broker.Admin.SDK.UserZonePreference object.

Related Links

Start-BrokerCatalogPvdImagePrepare

March 11, 2024

This cmdlet is no longer supported

Syntax

```
Start-BrokerCatalogPvdImagePrepare
      [-InputObject] <Catalog[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

PvD is no longer supported, use Citrix App Layering instead.

Examples

Parameters

-InputObject

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

Outputs

None

By default, this cmdlet returns no output.

Related Links

Start-BrokerDesktopGroupRebootCycle

March 11, 2024

Creates and starts a reboot cycle for each specified desktop group.

Syntax

```
Start-BrokerDesktopGroupRebootCycle
     [-InputObject] <DesktopGroup[]>
     -RebootDuration <Int32>
     [-RestrictToTagUid <Int32>]
     [-WarningDuration <Int32>]
     [-WarningTitle <String>]
     [-WarningMessage <String>]
     [-WarningRepeatInterval <Int32>]
     [-IgnoreMaintenanceMode <Boolean>]
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```


Description

The Start-BrokerDesktopGroupRebootCycle cmdlet is used to create and start a reboot cycle for specified desktop groups.

The functionality offered is similar to that of New-BrokerRebootSchedule but the resulting reboot cycles execute once as defined by the command parameters rather than on a repeating schedule.

Examples

EXAMPLE 1

Starts a new reboot cycle for the desktop group called “SampleGroup”. Each reboot cycle has a duration of six hours. Fifteen minutes prior to rebooting a machine, the message “Save your work”is displayed in each active user session. Only machines tagged with the tag having UID 2 will be rebooted.

```
Get-BrokerDesktopGroup "SampleGroup" | Start-
    BrokerDesktopGroupRebootCycle -RebootDuration 240 -WarningMessage "
    Save your work" -WarningDuration 15 -RestrictToTagUid 2
```

Parameters

-InputObject

Creates a reboot cycle for the specified desktop groups. Groups can be specified using UID values, name values (including wildcards) or desktop group SDK objects.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-RebootDuration

Approximate maximum duration in minutes over which the reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RestrictToTagUid

If set to a Tag UID, only machines that have this tag will be rebooted.

Type:	Int32
Position:	Named
Default value:	Null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningDuration

Time in minutes prior to a machine reboot at which a warning message is displayed in all user sessions on that machine. If the warning duration value is zero then no message is displayed.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Number of minutes to wait before showing the reboot warning message again.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-IgnoreMaintenanceMode

Option to reboot machines in maintenance mode.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

Desktop groups may be specified through pipeline input. The groups can be specified using UID values, name values (including wildcards) or desktop group objects

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Start-BrokerRebootCycle
- Stop-BrokerRebootCycle
- Get-BrokerRebootCycle

Start-BrokerMachinePvdImagePrepare

March 11, 2024

This cmdlet is no longer supported

Syntax

```
Start-BrokerMachinePvdImagePrepare
     [-InputObject] <Machine[]>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```


Description

PvD is no longer supported, use Citrix App Layering instead.

Examples

Parameters

-InputObject

Type:	Machine[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Machine

Outputs

None

By default, this cmdlet returns no output.

Related Links

Start-BrokerNaturalDesktopGroupRebootCycle

March 11, 2024

Reboots all machines from the specified desktop group when they are not in use.

Syntax

```
Start-BrokerNaturalDesktopGroupRebootCycle
     [-InputObject] <DesktopGroup[]>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Creating a natural reboot cycle for a desktop group ensures that all machines in the group are running the most recent image for the group.

The machines are rebooted in a non-disruptive manner, allowing machines that are in use to continue working and be restarted only after they become idle.

Examples

EXAMPLE 1

The above code applies a natural reboot cycle on desktop group with Id 1

```
$dg = Get-BrokerDesktopGroup -Uid 1
Start-BrokerNaturalDesktopGroupRebootCycle -InputObject $dg
```

Parameters

-InputObject

Reboots all machines from this input desktop group. The desktop groups can be specified using UID values, name values (including wildcards) or desktop group objects.

Type:	DesktopGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.DesktopGroup

Desktop Groups may be specified through pipeline input. The desktop groups can be specified using UID values, name values (including wildcards) or desktop groups objects

Outputs

None

By default, this cmdlet returns no output.

Notes

Natural reboot cycles do not apply to non-power managed machines or multi-session desktop groups

Related Links

- Start-BrokerRebootCycle
- Start-BrokerDesktopGroupRebootCycle
- Start-BrokerNaturalRebootCycle

Start-BrokerNaturalRebootCycle

March 11, 2024

Reboots all machines from the specified catalog when they are not in use.

Syntax

```
Start-BrokerNaturalRebootCycle
      [-InputObject] <Catalog[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Creating a natural reboot cycle for a catalog ensures that all machines in the catalog are running the most recent image for the catalog.

The machines are rebooted in a non-disruptive manner, allowing machines that are in use to continue working and be restarted only after they become idle.

Examples

EXAMPLE 1

The above code applies a natural reboot cycle on catalog with Id 1

```
$c = Get-BrokerCatalog -Uid 1
Start-BrokerNaturalRebootCycle -InputObject $c
```

Parameters

-InputObject

Reboots all machines from this input catalog. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects.

Type:	Catalog[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

Catalogs may be specified through pipeline input. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects

Outputs

None

By default, this cmdlet returns no output.

Notes

Natural reboot cycles do not apply to non-power managed and multi session catalogs

Related Links

- Start-BrokerRebootCycle
- Start-BrokerDesktopGroupRebootCycle
- Start-BrokerNaturalDesktopGroupRebootCycle

Start-BrokerRebootCycle

March 11, 2024

Creates and starts a reboot cycle for each desktop group that contains machines from the specified catalog.

Syntax

```
Start-BrokerRebootCycle
     [-InputObject] <Catalog[]>
     -RebootDuration <Int32>
     [-WarningDuration <Int32>]
     [-WarningTitle <String>]
     [-WarningMessage <String>]
     [-WarningRepeatInterval <Int32>]
```



```
8       [-LoggingId <Guid>]
9       [<CitrixCommonParameters>]
10      [<CommonParameters>]
```

Description

The Start-BrokerRebootCycle cmdlet is used to create and start a reboot cycle for each desktop group that contains machines from the specified catalog. For a given desktop group, only the machines from the target catalog are rebooted and any machines from other catalogs are not rebooted.

Creating a reboot cycle for catalog ensures that all machines in the catalog are running the most recent image for the catalog.

Examples

EXAMPLE 1

Starts a new reboot cycle for each desktop group containing machines from the catalog "SampleCatalog". Each reboot cycle has a duration of six hours. Fifteen minutes prior to rebooting a machine, the message "Save your work"is displayed in each active user session.

```
1 Get-BrokerCatalog -Name "SampleCatalog" | Start-BrokerRebootCycle -
      RebootDuration 240 -WarningMessage "Save your work" -WarningDuration
       15
```

EXAMPLE 2

Starts a new reboot cycle for each desktop group containing machines from the catalog "SampleCatalog". Fifteen, ten and five minutes prior to rebooting each machine, the message "Rebooting in %m% minutes."will be displayed in each user session with the pattern '%m%'replaced with the number of minutes until the reboot.

```
1 Get-BrokerCatalog -Name "SampleCatalog" | Start-BrokerRebootCycle -
      WarningMessage "Rebooting in %m% minutes." -WarningDuration 15 -
      WarningRepeatInterval 5
```

Parameters

-InputObject

Creates a reboot cycle for each desktop group that contains machines from this input catalog SDK object. The catalogs can be specified using UID values, name values (including wildcards) or catalog

objects.

Type:	Catalog[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-RebootDuration

Approximate maximum duration in minutes over which the reboot cycle runs.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningDuration

Time in minutes prior to a machine reboot at which a warning message is displayed in all user sessions on that machine. If the warning duration value is zero then no message is displayed. In some cases the time required to process a reboot cycle may exceed the RebootDuration time by up to the WarningDuration value; Citrix recommends that the WarningDuration is kept small relative to the RebootDuration value.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-WarningTitle

The window title used when showing the warning message in user sessions on a machine scheduled for reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningMessage

Warning message displayed in user sessions on a machine scheduled for reboot. If the message is blank then no message is displayed. The optional pattern '%m%'is replaced by the number of minutes until the reboot.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningRepeatInterval

Number of minutes to wait before showing the reboot warning message again.

Type:	Int32
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Catalog

Catalogs may be specified through pipeline input. The catalogs can be specified using UID values, name values (including wildcards) or catalog objects

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Start-BrokerDesktopGroupRebootCycle
- Stop-BrokerRebootCycle
- Get-BrokerRebootCycle

Start-BrokerSessionRecording

March 11, 2024

Starts recording the specified session(s).

Syntax

```
Start-BrokerSessionRecording
      [-Sessions] <Session[]>
      [-NotifyUser <Boolean>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Start-BrokerSessionRecording
      [-User] <User>
      [-NotifyUser <Boolean>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Starts recording the specified session(s).

Only active sessions using the HDX protocol can be recorded. An error appears when unqualified sessions (non-existent, inactive, or non-HDX) are specified for recording. No warning appears when you trigger an action to record sessions that are already being recorded.

Examples

EXAMPLE 1

Starts recording the session Uid 10,11,12 without notifying the user.

```
1 Start-BrokerSessionRecording -Sessions 10,11,12 -NotifyUser 0
```

EXAMPLE 2

Starts recording all sessions of user Geffrey in the domain named MYDOMAIN and notifies user Geffrey.

```
1 Start-BrokerSessionRecording -User MYDOMAIN\Geffrey -NotifyUser 1
```

Parameters

-Sessions

Specifies the session(s) to be recorded.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-User

Specifies the user whose session is to be recorded.

Type:	User
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-NotifyUser

Determines whether to notify the end user of the recording activity.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The session(s) to be recorded.

Citrix.Broker.Admin.SDK.User

The user whose sessions are to be recorded.

Outputs

None

By default, this cmdlet returns no output.

Notes

Sessions can be passed using the Sessions parameter as session objects or session Uids. A user can be passed using the User parameter as a user object or a user name.

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the Delivery Controller and the VDA, or if bad arguments are passed to the cmdlet itself, or if the VDA cannot successfully execute the operation.

The Delivery Controller can fail to invoke the operation if the VDA is not in an appropriate state or if there are problems with communicating with the VDA. When an operation is invoked, the Delivery Controller detects and reports whether the operation is initiated successfully on the VDA. However, because the operation is non-blocking, the Delivery Controller does not detect or report whether or not the operation ultimately succeeds after the operation is initiated successfully.

Operation failures are reported through the Broker SDK error handling mechanism (see about_Broker_ErrorHandli
In the event of errors, the SdkErrorRecord error status code is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successful operations.

The SdkErrorRecord message displays the number of attempted, failed, and successful operations in the following format:

"Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:<OperationsSucceededCount>"

Related Links

- Get-BrokerSession
- Get-BrokerSessionRecordingStatus
- Stop-BrokerSessionRecording

Stop-BrokerRebootCycle

March 11, 2024

Cancels the specified reboot cycle.

Syntax

```
Stop-BrokerRebootCycle
    [-InputObject] <RebootCycle[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

The Stop-BrokerRebootCycle cmdlet is used to cancel the specified reboot cycle.

Examples

EXAMPLE 1

Cancels every reboot cycle for the catalog that has the Uid of 7.

```
1  Get-BrokerRebootCycle -CatalogUid 7 | Stop-BrokerRebootCycle
```

Parameters

-InputObject

Cancels this reboot cycle.

Type:	RebootCycle[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.RebootCycle

Reboot cycles may be specified through pipeline input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-BrokerRebootCycle
- Start-BrokerRebootCycle
- Start-BrokerDesktopGroupRebootCycle

Stop-BrokerSession

March 11, 2024

Stop or log off a session.

Syntax

```
Stop-BrokerSession
    [-InputObject] <Session[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Stops or logs off sessions.

Examples

EXAMPLE 1

Stops all sessions for the user MyDomain\MyAccount.

```
Get-BrokerSession -UserName MyDomain\MyAccount | Stop-BrokerSession
```

EXAMPLE 2

Stops the session on MyMachine.

```
$desktop = Get-BrokerDesktop -DNSName MyMachine.MyDomain.com
Stop-BrokerSession $desktop.SessionUid
```

EXAMPLE 3

Stop sessions that have been disconnected for more than one day.

```
Get-BrokerSession -Filter {
  SessionState -eq 'Disconnected' -and SessionStateChangeTime -lt '-1'
      }
  | Stop-BrokerSession
```


EXAMPLE 4

Trap and display error information.

```
trap  [Citrix.Broker.Admin.SDK.SdkOperationException]
{

  write $("Exception name = " + $_.Exception.GetType().FullName)
  write $("SdkOperationException.Status = " + $_.Exception.Status)
  write $("SdkOperationException.ErrorData=")
  $_.Exception.ErrorData

  write $("SdkOperationException.InnerException = " + $_.Exception.
     InnerException)
  $_.Exception.InnerException
  continue
 }


Stop-BrokerSession -InputObject 10,11,12
```

Parameters

-InputObject

Identifies the session(s) to terminate. This can be expressed as either a session Uid or a session object.

Type:	Session[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The sessions to stop can be piped into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the controller and the machine, if bad arguments are passed to the cmdlet itself or if the machine cannot successfully execute the operation.

The transient nature of sessions means that the list of session objects or UIDs supplied to Stop-BrokerSession could consist of valid and invalid sessions. Invalid sessions are detected and disregarded and the stop session operation is invoked on the valid sessions.

The system can fail to invoke the operation if the machine is not in an appropriate state or if there are problems in communicating with the machine. When an operation is invoked the system detects if the operation was initiated successfully or not by the machine. As this operation is non-blocking the system doesn't detect or report whether the operation ultimately succeeded or failed after its successful initialization on the machine.

Operation failures are reported through the broker SDK error handling mechanism (see about_Broker_ErrorHandli
In the event of errors the SdkErrorRecord error status is set to SessionOperationFailed and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount - The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successfully executed operations.
- UnresolvedSessionFailuresCount - The number of operations that failed due to invalid sessions being supplied.
- OperationInvocationFailuresCount - The number of operations that failed because they could not be invoked on the desktop.
- DesktopExecutionFailuresCount - The number of operations that failed because they could not be successfully executed by the desktop.

The SdkErrorRecord message will also display the number of attempted, failed and successful operations in the following format:

"Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:<OperationsSucceededCount>"

Related Links

- Disconnect-BrokerSession
- Get-BrokerSession

Stop-BrokerSessionRecording

March 11, 2024

Stops recording the specified session(s).

Syntax

```
Stop-BrokerSessionRecording
    [-Sessions] <Session[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Stop-BrokerSessionRecording
    [-User] <User>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Stops recording the specified session(s).

Examples

EXAMPLE 1

Stops recording the session Uid 10,11,12.

```
Stop-BrokerSessionRecording -Sessions 10,11,12
```

EXAMPLE 2

Stops recording all sessions of user Administrator in the domain named MYDOMAIN.

```
Stop-BrokerSessionRecording -User MYDOMAIN\Administrator
```

Parameters

-Sessions

Specifies the session(s) to stop recording.

Type:	Session[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-User

Specifies the user whose sessions to stop recording.

Type:	User
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Broker.Admin.SDK.Session

The session(s) to stop recording.

Citrix.Broker.Admin.SDK.User

The user whose sessions to stop recording.

Outputs

None

By default, this cmdlet returns no output.

Notes

Sessions can be passed using the Sessions parameter as session objects or session Uids. A user can be passed using the User parameter as a user object or a user name. Either the Sessions or User parameter can be used at a time.

This operation is non-blocking and returns before it completes. The operation, however, is unlikely to fail unless there are communication problems between the Delivery Controller and the VDA, or if bad arguments are passed to the cmdlet itself, or if the VDA cannot successfully execute the operation.

The Delivery Controller can fail to invoke the operation if the VDA is not in an appropriate state or if there are problems with communicating with the VDA. When an operation is invoked, the Delivery Controller detects and reports whether the operation is initiated successfully on the VDA. However, because the operation is non-blocking, the Delivery Controller does not detect or report whether or not the operation ultimately succeeds after the operation is initiated successfully.

Operation failures are reported through the Broker SDK error handling mechanism (see about_Broker_ErrorHandli
In the event of errors, the SdkErrorRecord error status code is set to SessionOperationFailed, and its error data dictionary is populated with the following entries:

- OperationsAttemptedCount: The number of operations attempted.
- OperationsFailedCount - The number of failed operations.
- OperationsSucceededCount - The number of successful operations.

The SdkErrorRecord message displays the number of attempted, failed, and successful operations in the following format:

“Session operation error - attempted:<OperationsAttemptedCount>, failed:<OperationsFailedCount>, succeeded:<OperationsSucceededCount>”

Related Links

- Get-BrokerSession
- Get-BrokerSessionRecordingStatus
- Start-BrokerSessionRecording

Test-BrokerAccessPolicyRuleNameAvailable

March 11, 2024

Determine whether the proposed AccessPolicyRule Name is available for use.

Syntax

```
Test-BrokerAccessPolicyRuleNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed AccessPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name “Test1”is available.


```
Test-BrokerAccessPolicyRuleNameAvailable -Name Test1
```

EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerAccessPolicyRuleNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The AccessPolicyRule Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerAccessPolicyRule
- New-BrokerAccessPolicyRule
- Rename-BrokerAccessPolicyRule

Test-BrokerAppAssignmentPolicyRuleNameAvailable

March 11, 2024

Determine whether the proposed AppAssignmentPolicyRule Name is available for use.

Syntax

```
Test-BrokerAppAssignmentPolicyRuleNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed AppAssignmentPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerAppAssignmentPolicyRuleNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerAppAssignmentPolicyRuleNameAvailable @("Test1","Test2","
    Test3")
```

Parameters

-Name

The AppAssignmentPolicyRule Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerAppAssignmentPolicyRule
- New-BrokerAppAssignmentPolicyRule
- Rename-BrokerAppAssignmentPolicyRule

Test-BrokerAppEntitlementPolicyRuleNameAvailable

March 11, 2024

Determine whether the proposed AppEntitlementPolicyRule Name is available for use.

Syntax

```
Test-BrokerAppEntitlementPolicyRuleNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed AppEntitlementPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerAppEntitlementPolicyRuleNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerAppEntitlementPolicyRuleNameAvailable @("Test1","Test2","
    Test3")
```

Parameters

-Name

The AppEntitlementPolicyRule Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerAppEntitlementPolicyRule
- New-BrokerAppEntitlementPolicyRule
- Rename-BrokerAppEntitlementPolicyRule

Test-BrokerApplicationGroupNameAvailable

March 11, 2024

Determine whether the proposed ApplicationGroup Name is available for use.

Syntax

```
Test-BrokerApplicationGroupNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed ApplicationGroup Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerApplicationGroupNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerApplicationGroupNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The ApplicationGroup Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerApplicationGroup
- New-BrokerApplicationGroup
- Rename-BrokerApplicationGroup

Test-BrokerApplicationNameAvailable

March 11, 2024

Determine whether the proposed Application Name is available for use.

Syntax

```
Test-BrokerApplicationNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed Application Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerApplicationNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerApplicationNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The Application Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerApplication
- New-BrokerApplication
- Rename-BrokerApplication

Test-BrokerAssignmentPolicyRuleNameAvailable

March 11, 2024

Determine whether the proposed AssignmentPolicyRule Name is available for use.

Syntax

```
Test-BrokerAssignmentPolicyRuleNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed AssignmentPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerAssignmentPolicyRuleNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerAssignmentPolicyRuleNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The AssignmentPolicyRule Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerAssignmentPolicyRule
- New-BrokerAssignmentPolicyRule
- Rename-BrokerAssignmentPolicyRule

Test-BrokerCatalogNameAvailable

March 11, 2024

Determine whether the proposed Catalog Name is available for use.

Syntax

```
Test-BrokerCatalogNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed Catalog Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerCatalogNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerCatalogNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The Catalog Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of “True”indicates the Name is available for use, and “False”if it is not available.

Related Links

- Get-BrokerCatalog
- New-BrokerCatalog
- Rename-BrokerCatalog

Test-BrokerDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Broker Service.

Syntax

```
Test-BrokerDBConnection
    [-DBConnection] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-BrokerDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-BrokerDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Broker Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Broker SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1  Test-BrokerDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
       Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Broker Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-BrokerDBConnection cmdlet returns an object describing the status of the selected Broker Service instance that would result if the connection string were used with the Set-BrokerDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-BrokerDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Broker Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Broker Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Broker Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Broker Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Broker Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Broker Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_Broker_Concepts
- Get-BrokerServiceStatus
- Get-BrokerDBConnection
- Set-BrokerDBConnection

Test-BrokerDesktopGroupNameAvailable

March 11, 2024

Determine whether the proposed DesktopGroup Name is available for use.

Syntax

```
Test-BrokerDesktopGroupNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed DesktopGroup Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name “Test1”is available.

```
Test-BrokerDesktopGroupNameAvailable -Name Test1
```

EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerDesktopGroupNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The DesktopGroup Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerDesktopGroup
- New-BrokerDesktopGroup
- Rename-BrokerDesktopGroup

Test-BrokerEntitlementPolicyRuleNameAvailable

March 11, 2024

Determine whether the proposed EntitlementPolicyRule Name is available for use.

Syntax

```
Test-BrokerEntitlementPolicyRuleNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed EntitlementPolicyRule Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name “Test1”is available.

```
Test-BrokerEntitlementPolicyRuleNameAvailable -Name Test1
```

EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerEntitlementPolicyRuleNameAvailable @("Test1","Test2","Test3"
    )
```

Parameters

-Name

The EntitlementPolicyRule Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerEntitlementPolicyRule
- New-BrokerEntitlementPolicyRule
- Rename-BrokerEntitlementPolicyRule

Test-BrokerLicenseServer

March 11, 2024

Tests whether or not a license server can be used by the broker.

Syntax

```
Test-BrokerLicenseServer
    [-ComputerName] <String>
    [[-Port] <Int32>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether or not a given license server can be used by the broker.

Examples

EXAMPLE 1

Tests whether or not the license server “machine.domain”with port number 1234 can be used.

```
Test-BrokerLicenseServer -LicenseServerAddress "machine.domain" 1234
```

EXAMPLE 2

Tests whether or not the license server “machine.domain”with port number 2700 can be used.

```
Test-BrokerLicenseServer -LicenseServerAddress "machine.domain"
```

Parameters

-ComputerName

The name of the license server to test (machine.domain).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Port

The port number to use on the server.

Type:	Int32
Position:	3
Default value:	27000
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

Test-BrokerLicenseServer returns:

- 'Compatible'- the server is a compatible license server that can be used.
- 'Incompatible'- the server is an incompatible license server that can't be used.

- 'Inaccessible'- the server cannot be accessed. The server may be down, unreachable, or non-existent.
- 'InternalError - the server can't be used due to an internal error. A required licensing component on the server may not be installed, configured, or working correctly.

Related Links

- Get-BrokerSite
- Set-BrokerSite

Test-BrokerMachineNameAvailable

March 11, 2024

Determine whether the proposed Machine MachineName is available for use.

Syntax

```
Test-BrokerMachineNameAvailable
    [-MachineName] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed Machine MachineName is available for use. It returns a record for each MachineName indicating the availability of that MachineName, with $true indicating that the MachineName is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the MachineName "Test1"is available.

```
Test-BrokerMachineNameAvailable -MachineName Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerMachineNameAvailable @("Test1","Test2","Test3")
```

Parameters

-MachineName

The Machine MachineName to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the MachineName to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each MachineName specified. An availability of "True"indicates the MachineName is available for use, and "False"if it is not available.

Related Links

- Get-BrokerMachine
- New-BrokerMachine

Test-BrokerPowerTimeSchemeNameAvailable

March 11, 2024

Determine whether the proposed PowerTimeScheme Name is available for use.

Syntax

```
Test-BrokerPowerTimeSchemeNameAvailable
    [-Name] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed PowerTimeScheme Name is available for use. It returns a record for each Name indicating the availability of that Name, with $true indicating that the Name is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the Name "Test1"is available.

```
Test-BrokerPowerTimeSchemeNameAvailable -Name Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerPowerTimeSchemeNameAvailable @("Test1","Test2","Test3")
```

Parameters

-Name

The PowerTimeScheme Name to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the Name to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each Name specified. An availability of "True"indicates the Name is available for use, and "False"if it is not available.

Related Links

- Get-BrokerPowerTimeScheme
- New-BrokerPowerTimeScheme
- Rename-BrokerPowerTimeScheme

Test-BrokerRemotePCAccountNameAvailable

March 11, 2024

Determine whether the proposed RemotePCAccount OU is available for use.

Syntax

```
Test-BrokerRemotePCAccountNameAvailable
    [-OU] <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet checks whether proposed RemotePCAccount OU is available for use. It returns a record for each OU indicating the availability of that OU, with $true indicating that the OU is unused and available for use, or $false if it is not available.

Examples

EXAMPLE 1

Checks whether the OU "Test1"is available.

```
Test-BrokerRemotePCAccountNameAvailable -OU Test1
```


EXAMPLE 2

Checks whether each of the specified names is available.

```
Test-BrokerRemotePCAccountNameAvailable @("Test1","Test2","Test3")
```

Parameters

-OU

The RemotePCAccount OU to be tested.

Type:	String[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains the OU to test.

Outputs

Citrix.Broker.Admin.SDK.NameAvailability

The cmdlet returns a result for each OU specified. An availability of "True"indicates the OU is available for use, and "False"if it is not available.

Related Links

- Get-BrokerRemotePCAccount
- New-BrokerRemotePCAccount

Update-BrokerGpoPolicySetBlob

March 11, 2024

Force an immediate update of a policy set's data blob.

Syntax

```
Update-BrokerGpoPolicySetBlob
      -PolicySetGuid <Guid>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Update-BrokerGpoPolicySetBlob cmdlet is used to force an immediate conversion of the policy set's data stored in database tables to the compiled data blob for the policy set. Normally this is not necessary because a site service regularly converts the policy data in a policy set to a data blob at 5 minute intervals. Use this command to have the data updated immediately.

Examples

EXAMPLE 1

Update the policy blob data of a policy set.

```
Update-BrokerGpoPolicySetBlob -PolicyGuid ([Guid]"abcdef12-...")
```


Parameters

-PolicySetGuid

The GUID of the policy set to be updated. Only a policy set of type DeliveryGroupPolicies can be updated. If a policy set of another type is specified or if a non-existent policy set is specified, this command does nothing.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

This cmdlet does not generate any output.

Related Links

- about_Broker_GroupPolicy

Update-BrokerImportedFTA

March 11, 2024

Imports or updates all of the file type associations for the

specified worker.

Syntax

```
Update-BrokerImportedFTA
      -DesktopUids <Int32[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Imports or updates the file type associations from a specified worker machine.

File type association associates a file extension (such as ".txt") with an application (such as Notepad). In a Citrix environment file type associations on a user device can be configured so that when an user clicks on a document it launches the appropriate published application. This is known as "content redirection".

Imported file type associations are different from configured file type associations. Imported file type associations are lists of known file type associations for a given desktop group. Configured file type associations are those that are actually associated with published applications for the purposes of content redirection.

Initially the system is not aware of any extensions, and they must be imported by the Citrix administrator. To import or update file type associations from a worker machine, two criteria must be met:

- The worker machine must not be in use
- The worker machine must be in maintenance mode

For more information about putting a worker machine in maintenance mode, see the Set-BrokerPrivateDesktop and Set-BrokerSharedDesktop cmdlets.

Imported file type associations are grouped together based on the desktop group of the machine from which they were imported. All file types for a desktop group are deleted. There is no mechanism for deleting a subset imported file type associations for a specific desktop group.

If file type associations are imported more than once for a desktop group, for example, if this cmdlet is run twice for two workers belonging to the same desktop group, all existing imported file type associations for that desktop group are deleted and imported again.

Examples

EXAMPLE 1

Gets an object for the worker machine named

“Worker1”in the “ACME”domain, and ensures no users can

connect to it, before importing the file type associations from that

desktop and re-enabling it.

```
$desktop = Get-BrokerSharedDesktop -MachineName "ACME\Worker1"
Set-BrokerSharedDesktop $desktop -InMaintenanceMode $true
Update-BrokerImportedFTA -DesktopUids $desktop.Uid
Set-BrokerSharedDesktop $desktop -InMaintenanceMode $false
```

Parameters

-DesktopUids

Imports or updates the file type associations from the

specified desktop. The desktop must belong to a desktop group of the

Private or Shared desktop kind.

Type:	Int32[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Int32[]

An array of Uids for desktops can be supplied as input.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_Broker_Applications
- Get-BrokerImportedFTA
- Remove-BrokerImportedFTA

Update-BrokerLocalLeaseCache

March 11, 2024

Flushes the local lease cache.

Syntax

```
Update-BrokerLocalLeaseCache
      [-Workers]
      [-Applications]
      [-Icons]
      [-Desktops]
      [-Leases]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Removes all local cached lease data and any state information stored in the registry.

Examples

EXAMPLE 1

Flushes the local lease cache for all objects and deletes any state information stored in the registry.

```
Update-BrokerLocalLeaseCache
```

EXAMPLE 2

Flushes the local lease cache for all workers and deletes any state information stored in the registry.


```
Update-BrokerLocalLeaseCache -Workers
```

Parameters

-Workers

Removes all locally cached workers on the controller. The worker cache will be repopulated from the current site database contents after a short delay.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Applications

Removes all locally cached applications on the controller. The application cache will be repopulated from the current site database contents after a short delay.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Icons

Removes all locally cached icons on the controller. The icon cache will be repopulated from the current site database contents after a short delay.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Desktops

Removes all locally cached desktops on the controller. The desktop cache will be repopulated from the current site database contents after a short delay.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Leases

Removes all locally cached leases on the controller. The lease cache will be repopulated from the current site database contents after a short delay.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high level operation that this cmdlet call forms a part of. Desktop Studio and Desktop Director typically create High Level Operations. PowerShell scripts can also wrap a series of cmdlet calls in a High Level Operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

The local cache for lease and other data like worker, desktop, application and icon information is removed and will be repopulated from the current site database contents after a short delay.

Related Links

- Remove-BrokerLease

Update-BrokerNameCache

March 11, 2024

Performs administrative operations on the user/group and machine name cache.

Syntax

```
Update-BrokerNameCache
      [-Machines]
      [-Users]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Update-BrokerNameCache
      [-Machines]
      [-Users]
      [-Purge]
      [-UnusedFor <TimeSpan>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Update-BrokerNameCache
      -Machine <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Update-BrokerNameCache
      -User <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Triggers an immediate asynchronous refresh of the name cache, or when used with the -Purge parameter removes cache entries that are unreferenced and have been unused for a period of time.

The Broker Service maintains a cache of the names of users/groups and machines in use by the site. By default, name information is obtained periodically from Active Directory and the cache refreshed automatically.

Triggering a cache refresh with this cmdlet ensures up-to-date name information is present in the cache after user/group or machine accounts are known to have changed and you need to see those changes immediately instead of waiting for the periodic automatic refresh.

Removing (purging) entries relating to no longer used user/group or machine accounts is useful to stop requests being made to Active Directory to refresh the names relating to those items.

For users/groups, the following name information is cached:

Windows name (DOMAIN\user)

User Principal Name or 'UPN'(user@upndomain)

Full Name or 'Common Name'(typically a user's full name)

For machines, the following name information is cached:

Windows name (DOMAIN\machine)

DNS name (machine.dnsdomain)

Examples

EXAMPLE 1

Triggers an immediate asynchronous refresh of all machine name information held within the name cache.

```
1  Update-BrokerNameCache -Machines
```

EXAMPLE 2

Triggers an immediate asynchronous refresh of all machine and user name information held within the name cache.

```
1  Update-BrokerNameCache -Machines -Users
```


Parameters

-Purge

Removes unused entries from the cache rather than refreshing the name data. Entries are only removed if they are not configured or referenced in any way within the site.

In addition, user entries are only removed if the user has not logged in for at least 90 days. Similarly machine entries are only removed if the machine has not registered with the site over the same period. The 90 day default can be overridden using the -UnusedFor parameter.

While removing genuinely unused cache entries is desirable to reduce the requests made to Active Directory, there is additional cost in repopulating the cache if a user/group or machine account is later used again after having been removed. Thus it is not recommended to routinely use very short intervals with -UnusedFor.

If a machine entry is removed from the cache and later used again, the Uid value associated with that machine changes (seen in the output of Get-BrokerMachine).

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Machine

Triggers an asynchronous refresh of the cached machine name information for the specified machine. The machine can be specified by SID or by its name as currently known to the Citrix Broker service (for example as shown by Get-BrokerMachine).

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-User

Triggers an asynchronous refresh of the cached user name information for the specified user. The user can be specified by SID or by its name as currently known to the Citrix Broker service (for example as shown by Get-BrokerUser).

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Machines

Triggers an asynchronous refresh of all cached machine name information, or with the -Purge parameter removes unused machine entries from the cache.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Users

Triggers an asynchronous refresh of all cached user/group name information, or with the -Purge parameter removes unused user/group entries from the cache.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UnusedFor

Specifies the minimum period over which user or machine cache entries must have been unused before they can be be removed with the -Purge option.

Type:	TimeSpan
Position:	Named
Default value:	90 days
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

For some user accounts, for example, the built-in domain administrator, the UPN and/or Full Name values may not be available because they are not typically specified within Active Directory.

For group accounts, UPN and Full Name values are not available because they are not applicable or not specified within Active Directory.

The DNS name information for a machine is obtained from Active Directory and not from the DNS subsystem. If a machine has only recently been configured, the DNS information may not be available initially.

Related Links

Update-BrokerScopeCache

March 11, 2024

Synchronise the scope information with the Delegated Administration service

Syntax

```
Update-BrokerScopeCache
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Force an immediate synchronisation of the scope information held by the Delegated Administration

service with the cached copy held by the Broker service. The scope data would be eventually synchronised anyway, but this

cmdlet can be used when the synchronisation is needed immediately.

Examples

EXAMPLE 1

Ensures the broker cache is up to date after a change to the scopes configured in the site

```
New-AdminScope -Name 'NewScopeName'
Update-BrokerScopeCache
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

about_ConfigConfigurationSnapIn

March 11, 2024

Topic

about_ConfigConfigurationSnapin

Short Description

The Configuration service PowerShell snap-in provides administrative functions for the Configuration service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Config'.

Long Description

The Configuration service PowerShell snap-in enables both local and remote administration of the Configuration service. It provides facilities to store details about other services that are used in the XenDesktop deployment.

All the services in the XenDesktop deployment use the Configuration service as a directory to locate other services with which they need to communicate. The directory publishes a list of all the services, the communication types they accept, and the facilities they offer.

The snap-in provides the following main entities:

Site Metadata

Metadata for the entire XenDesktop deployment. This metadata is not used directly by any of the XenDesktop services, but can be used by third parties to store information for other purposes.

Registered Service Instances

Service instance items that have been retrieved from the available services in the XenDesktop deployment and held in a directory in the Configuration service. Each physical service can host a collection of service instances to provide different facilities.

Service Groups

A collection of service instances that are considered equivalent. Service instances that are registered in the same service group provide the same state and offer the same functionality.

Only one service group of each type is supported. For example, there must not be more than one service group with a ServiceType of 'Config'.

Site and Features

The configuration site is a top-level, logical representation of the XenDesktop site, from the perspective of the configuration services running within the site.

about_ConfigConfigurationSnapIn

March 11, 2024

Topic

about_ConfigConfigurationSnapin

Short Description

The Configuration service PowerShell snap-in provides administrative functions for the Configuration service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Config'.

Long Description

The Configuration service PowerShell snap-in enables both local and remote administration of the Configuration service. It provides facilities to store details about other services that are used in the XenDesktop deployment.

All the services in the XenDesktop deployment use the Configuration service as a directory to locate other services with which they need to communicate. The directory publishes a list of all the services, the communication types they accept, and the facilities they offer.

The snap-in provides the following main entities:

Site Metadata

Metadata for the entire XenDesktop deployment. This metadata is not used directly by any of the XenDesktop services, but can be used by third parties to store information for other purposes.

Registered Service Instances

Service instance items that have been retrieved from the available services in the XenDesktop deployment and held in a directory in the Configuration service. Each physical service can host a collection of service instances to provide different facilities.

Service Groups

A collection of service instances that are considered equivalent. Service instances that are registered in the same service group provide the same state and offer the same functionality.

Only one service group of each type is supported. For example, there must not be more than one service group with a ServiceType of 'Config'.

Site and Features

The configuration site is a top-level, logical representation of the XenDesktop site, from the perspective of the configuration services running within the site.

about_Config_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also

search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z"}
$d = [DateTime]"2010-08-23T12:30:00.0Z"

Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2'} # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30'} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30'} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*"}
Get-<Noun> -Filter { Users -contains "Fred*"}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'

Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects

with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries. It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-ConfigRegisteredServiceInstanceMetadata

March 11, 2024

Adds metadata on the given ServiceInstance.

Syntax

```
Add-ConfigRegisteredServiceInstanceMetadata
   [-ServiceInstanceUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigRegisteredServiceInstanceMetadata
   [-ServiceInstanceUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigRegisteredServiceInstanceMetadata
   [-InputObject] <ServiceInstance[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigRegisteredServiceInstanceMetadata
   [-InputObject] <ServiceInstance[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Provides the ability for additional custom data to be stored against given ServiceInstance objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ConfigRegisteredServiceInstanceMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the ServiceInstance with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Add-ConfigRegisteredServiceInstanceMetadata -ServiceInstanceUid 4
   CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-ServiceInstanceUid

Id of the ServiceInstance

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceInstance[]

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the ServiceInstance specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.Metadata

Add-ConfigRegisteredServiceInstanceMetadata returns an array of objects containing the new definition of the metadata.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigRegisteredServiceInstanceMetadata
- Remove-ConfigRegisteredServiceInstanceMetadata

Add-ConfigServiceGroupMetadata

March 11, 2024

Adds metadata on the given ServiceGroup.

Syntax

```
Add-ConfigServiceGroupMetadata
   [-ServiceGroupUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigServiceGroupMetadata
   [-ServiceGroupUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigServiceGroupMetadata
   [-InputObject] <ServiceGroup[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ConfigServiceGroupMetadata
   [-InputObject] <ServiceGroup[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given ServiceGroup objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ConfigServiceGroupMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the ServiceGroup with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.


```
Add-ConfigServiceGroupMetadata -ServiceGroupUid 4CECC26E-48E1-423F-A1F0
    -2A06DDD0805C -Name property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-ServiceGroupUid

Id of the ServiceGroup

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the ServiceGroup specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.Metadata

Add-ConfigServiceGroupMetadata returns an array of objects containing the new definition of the metadata.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigServiceGroupMetadata
- Remove-ConfigServiceGroupMetadata

Export-ConfigConfiguration

March 11, 2024

Obtains an XML document containing the configuration of the Site

Syntax

```
Export-ConfigConfiguration
       [-ExistingConfigLastChangeTime <String>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Examples

Parameters

-ExistingConfigLastChangeTime

The value of ConfigLastChangeTime in the site object of any configuration already held by the caller. If nothing has changed, an empty configuration will be returned with the “Updated”attribute set to false.

Type: String

Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

This cmdlet does not accept input

Outputs

String

The XML document

Related Links

Export-ConfigFeatureTable

March 11, 2024

Returns the current feature table.

Syntax

```
Export-ConfigFeatureTable
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Examples

EXAMPLE 1

Returns the current feature table.

```
Export-ConfigFeatureTable
<?xml version="1.0" encoding="utf-8">
<Products xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:
   xsd="http://www.w3.org/2001/XMLSchema" xmlns="FeatureTable.xsd">
  <Product Code="XDT" Name="XenDesktop">
    <Editions>
      <Edition>PLT</Edition>
      <Edition>ENT</Edition>
      <Edition>APP</Edition>
      <Edition>STD</Edition>
    </Editions>
    <DefaultEdition>PLT</DefaultEdition>
    <VersionToBurninDates>
      <VersionToBurninDate Version="7.0" BurninDate="2013.0522" />
    </VersionToBurninDates>
    <LicensingModels>
      <LicensingModel>Concurrent</LicensingModel>
      <LicensingModel>UserDevice</LicensingModel>
    </LicensingModels>
    <DefaultLicensingModel>UserDevice</DefaultLicensingModel>
    <Features>
      <Feature Name="CustomRole">
        <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
           >2013.0522</BurninDate>
        <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
           >2013.0522</BurninDate>
        <BurninDate ForProductEdition="APP" MustBe="AtLeast"
           >2013.0522</BurninDate>
      </Feature>
      <Feature Name="ConfigurationLogging">
        <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
           >2013.0522</BurninDate>
        <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
           >2013.0522</BurninDate>
        <BurninDate ForProductEdition="APP" MustBe="AtLeast"
           >2013.0522</BurninDate>
      </Feature>
      <Feature Name="SingleUserMode">
```



```
32          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
33          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
34        </Feature>
35        <Feature Name="MultiSessionDesktops">
36          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
37          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
38          <BurninDate ForProductEdition="APP" MustBe="AtLeast"
                >2013.0522</BurninDate>
39        </Feature>
40        <Feature Name="SingleSessionDesktops">
41          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
42          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
43          <BurninDate ForProductEdition="STD" MustBe="AtLeast"
                >2013.0522</BurninDate>
44        </Feature>
45        <Feature Name="MultiSessionApplications">
46          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
47          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
48          <BurninDate ForProductEdition="APP" MustBe="AtLeast"
                >2013.0522</BurninDate>
49        </Feature>
50        <Feature Name="SingleSessionApplications">
51          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
52          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
53          <BurninDate ForProductEdition="APP" MustBe="AtLeast"
                >2013.0522</BurninDate>
54        </Feature>
55        <Feature Name="HistoricalMonitorData">
56          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
57        </Feature>
58        <Feature Name="CloudHostedMachines">
59          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
60        </Feature>
61        <Feature Name="HdxInsightIntegration">
62          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
63        </Feature>
64        <Feature Name="RemotePC">
65          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
66          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
```



```
                >2013.0522</BurninDate>
67        </Feature>
68      </Features>
69    </Product>
70    <Product Code="MPS" Name="XenApp">
71      <Editions>
72        <Edition>PLT</Edition>
73        <Edition>ENT</Edition>
74      </Editions>
75      <DefaultEdition>PLT</DefaultEdition>
76      <VersionToBurninDates>
77        <VersionToBurninDate Version="7.0" BurninDate="2013.0522" />
78      </VersionToBurninDates>
79      <LicensingModels>
80        <LicensingModel>Concurrent</LicensingModel>
81      </LicensingModels>
82      <DefaultLicensingModel>Concurrent</DefaultLicensingModel>
83      <Features>
84        <Feature Name="CustomRole">
85          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
86          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
87        </Feature>
88        <Feature Name="ConfigurationLogging">
89          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
90          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
91        </Feature>
92        <Feature Name="SingleUserMode">
93        </Feature>
94        <Feature Name="MultiSessionDesktops">
95          <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
96          <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
97        </Feature>
98        <Feature Name="SingleSessionDesktops">
99        </Feature>
100       <Feature Name="MultiSessionApplications">
101         <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
102         <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
103       </Feature>
104       <Feature Name="SingleSessionApplications">
105         <BurninDate ForProductEdition="PLT" MustBe="AtLeast"
                >2013.0522</BurninDate>
106         <BurninDate ForProductEdition="ENT" MustBe="AtLeast"
                >2013.0522</BurninDate>
107       </Feature>
108       <Feature Name="HistoricalMonitorData">
```



```
109         </Feature>
110         <Feature Name="CloudHostedMachines">
111         </Feature>
112         <Feature Name="HdxInsightIntegration">
113         </Feature>
114         <Feature Name="RemotePC">
115         </Feature>
116       </Features>
117     </Product>
118   </Products>
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

Related Links

- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigConnectorAppliance

March 11, 2024

Gets Connector Appliances configured for this site

Syntax

```
Get-ConfigConnectorAppliance
    [-Uid <Guid>]
    [[-Name] <String>]
    [-ConnectorType <String>]
    [-Description <String>]
    [-IsHealthy <Boolean>]
    [-MachineAddress <String>]
    [-TenantId <Guid>]
    [-Uuid <Guid>]
    [-ZoneName <String>]
    [-ZoneUid <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves Connector Appliances matching the specified criteria. If no parameters specified, returns all known connectors

Examples

Parameters

-Name

Gets the connector appliances with the specified name

Type: String

Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Uid

Gets the connector appliances with the specified UID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ConnectorType

Gets the connector appliance types

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets the connector appliances with the specified description

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsHealthy

Gets the connector appliances with the specified healthy state

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineAddress

Gets the connector appliances with the specified machine address

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantId

Gets the connector appliances associated with the specified Tenant ID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Uuid

Gets the connector appliances with the specified uuid

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets the connector appliances located in the given zone specified by zone name

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets the connector appliances located in the given zone specified by zone UID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Get-ConfigDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Configuration Service.

Syntax

```
Get-ConfigDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected Configuration Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Configuration Service instance running on controller "controller1.mydomain.net".

```
1  Get-ConfigDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Configuration Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the ConfigurationService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigDBConnection
- Get-ConfigServiceStatus
- Test-ConfigDBConnection

Get-ConfigDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Configuration Service.

Syntax

```
Get-ConfigDBSchema
   [-ZoneUid <Guid>]
   [-DatabaseName <String>]
   [-ServiceGroupName <String>]
   [-ScriptType <ScriptTypes>]
```



```
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Configuration Service database schema, add a new Configuration service to an existing site, remove a Configuration service from a site, or create a database server logon for a Configuration service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Configuration service instance, otherwise the scripts relate to Configuration service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Configuration service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Configuration service roles

If ScriptType is Evict, the returned script contains:

- Removal of Configuration service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Configuration Service and copies it to a file called "C:\ConfigurationSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Configuration service schema.

```
Get-ConfigDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\ConfigurationSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Configuration service. This can be used when configuring a mirror server for use.

```
Get-ConfigDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    ConfigurationLogins.sql
```

Parameters

-ZoneUid

Allows adding a new Configuration Service instance to a specified Zone. If not specified the service instance is placed in the Primary zone. Can only be used with database script type Instance.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

Specifies the name of the database into which the new Configuration service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Configuration services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Configuration Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Configuration Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Configuration Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Configuration Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Configuration services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Configuration Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigDBConnection
- Test-ConfigDBConnection

Get-ConfigDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Configuration Service.

Syntax

```
Get-ConfigDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Configuration Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Configuration Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Configuration Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Configuration Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Configuration Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-ConfigServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-ConfigDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Configuration Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-ConfigDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Configuration Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Configuration services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-ConfigServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigInstalledDBVersion
- Get-ConfigServiceStatus
- Get-ConfigDBSchema

Get-ConfigEdgeServer

March 11, 2024

Gets Edge Servers configured for this site

Syntax

```
Get-ConfigEdgeServer
    [-Uid <Guid>]
    [[-Name] <String>]
    [-Description <String>]
    [-IsHealthy <Boolean>]
    [-MachineAddress <String>]
    [-Metadata <String>]
    [-Sid <String>]
    [-TenantId <Guid>]
    [-Uuid <Guid>]
    [-ZoneName <String>]
    [-ZoneUid <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves Edge Servers matching the specified criteria. If no parameters specified, returns all known Edge Servers

Examples

EXAMPLE 1

This command returns all edge servers in the zone named MyZone

```
Get-ConfigEdgeServer -ZoneName MyZone
```

EXAMPLE 2

This command returns the edge server with the specified SID value

```
Get-ConfigEdgeServer -Sid 21-3623811015-3361044348-30300820
```


Parameters

-Name

Gets the edge servers with the specified name

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Uid

Gets the edge servers with the specified UID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Gets the edge servers with the specified description

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-IsHealthy

Gets the edge servers with the specified healthy state

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineAddress

Gets the edge servers with the specified machine address

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Gets the edge servers with the specified SID value

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantId

Gets the edge servers associated with the specified Tenant ID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Uuid

Gets the edge servers with the specified uuid

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Gets the edge servers located in the given zone specified by zone name

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ZoneUid

Gets the edge servers located in the given zone specified by zone UID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Configuration.Sdk.EdgeServer

Get-ConfigEdgeServer returns an object for each matching Edge Server object.

Related Links

- New-ConfigEdgeServer
- Set-ConfigEdgeServer
- Rename-ConfigEdgeServer
- Remove-ConfigEdgeServer

Get-ConfigEnabledFeature

March 11, 2024

Lists features of the site that are enabled.

Syntax

```
Get-ConfigEnabledFeature
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Examples

EXAMPLE 1

Retrieves the list of enabled features for the site's current configuration.

```
Get-ConfigEnabledFeature
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

Related Links

- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Configuration Service.

Syntax

```
Get-ConfigInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Gets the current version number of the Citrix Configuration Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-ConfigDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Configuration Service database schema version number.

```
Get-ConfigInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Configuration Service database schema for which upgrade scripts are supplied.

```
Get-ConfigInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-ConfigInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Configuration

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigDBVersionChangeScript
- Get-ConfigDBSchema

Get-ConfigLicensingModel

March 11, 2024

Lists the supported licensing models.

Syntax

```
Get-ConfigLicensingModel
   -ProductCode <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Examples

EXAMPLE 1

Retrieves the list of supported licensing models for product code “XDT”.

```
Get-ConfigLicensingModel -ProductCode "XDT"
```

Parameters

-ProductCode

The product code

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

The list of supported licensing models for the specified product code.

Notes

The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by the Get-ConfigSite cmdlet contains the currently configured product code.

Related Links

- Get-ConfigProduct
- Get-ConfigSite
- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigLocalData

March 11, 2024

Gets the service local data.

Syntax

```
Get-ConfigLocalData
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Get-ConfigLocalData returns service-and-controller-specific local data. This information is not site-wide, but rather controller-specific. The Configuration Service currently stores the controller product version as local data. The overall site version used by Studio is an aggregate of the product versions from each controller.

Examples

EXAMPLE 1

Gets the service local data.

```
Get-ConfigLocalData

ControllerProductVersion
------------
7.40
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.DataModel.LocalData

Contains the ControllerProductVersion field

Related Links

- about_ConfigConfigurationSnapin

Get-ConfigProduct

March 11, 2024

Lists the site's supported product names and codes.

Syntax

```
Get-ConfigProduct
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Examples

EXAMPLE 1

Lists the supported products by name and code.

```
Get-ConfigProduct
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

Related Links

- Get-ConfigSite

- Set-ConfigSite
- about_ConfigConfigurationSnapin

Get-ConfigProductEdition

March 11, 2024

Lists the supported product editions.

Syntax

```
Get-ConfigProductEdition
    [-ProductCode] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Examples

EXAMPLE 1

Retrieves the list of supported editions for XenDesktop.

```
Get-ConfigProductEdition -ProductCode "XDT"
```

Parameters

-ProductCode

The product code

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

The list of supported licensing models for the specified product code.

Notes

The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by Get-ConfigSite cmdlet contains the currently configured product code.

Related Links

- Get-ConfigProduct
- Get-ConfigSite
- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigProductFeature

March 11, 2024

Lists the supported features.

Syntax

```
Get-ConfigProductFeature
   [-ProductCode] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Lists the supported features. Use the Get-ConfigEnabledFeature command to determine which features are currently enabled.

Examples

EXAMPLE 1

Retrieves the list of supported features for XenDesktop.

```
Get-ConfigProductFeature -ProductCode "XDT"
```

Parameters

-ProductCode

The product code

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

The list of supported licensing models for the specified product code.

Notes

The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by Get-ConfigSite cmdlet contains the currently configured product code.

Related Links

- Get-ConfigProduct
- Get-ConfigSite
- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigProductVersion

March 11, 2024

Lists the supported product versions.

Syntax

```
Get-ConfigProductVersion
   [-ProductCode] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Examples

EXAMPLE 1

Retrieves the list of supported versions for XenDesktop.

```
Get-ConfigProductVersion -ProductCode "XDS"
```

Parameters

-ProductCode

The product code

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

The list of supported licensing models for the specified product code.

Notes

The Get-ConfigProduct cmdlet lists the available product codes.

The site object returned by Get-ConfigSite cmdlet contains the currently configured product code.

Related Links

- Get-ConfigProduct
- Get-ConfigSite
- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigRegisteredServiceInstance

March 11, 2024

Gets the service instances that are registered in the directory.

Syntax

```
Get-ConfigRegisteredServiceInstance
    [-ServiceInstanceUid <Guid>]
    [-ServiceGroupUid <Guid>]
    [-ServiceGroupName <String>]
    [-ServiceType <String>]
    [-Address <String>]
    [-Binding <String>]
    [-Version <Int32>]
    [-ServiceAccountSid <String>]
    [-InterfaceType <String>]
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this cmdlet to retrieve the service instances currently registered with the Configuration Service that match the parameters supplied. If no parameters are supplied, all the service instances are returned.

Examples

EXAMPLE 1

Return all the service instances that are registered in the Configuration Service.

```
C:\>Get-ConfigRegisteredServiceInstance

Address             : http://MyServer.com:80/Citrix/MyContract/v1
Binding             : wcf_HTTP_kerb
InterfaceType       : SDK
Metadata            : {
  }

MetadataMap         : {
  }

ServiceAccount      : ENG\MyAccount$
ServiceAccountSid   : S-1-5-21-1155438255-2213498043-2452000591-1104
```



```
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8dc38b5a-3fbb-457c-b326-6c41c94c18d5
ServiceType        : MySnapIn
Version            : 1

Address            : http://MyServer.com:80/Citrix/MyContract/PeerAPI/
    v1
Binding            : wcf_HTTP_kerb
InterfaceType      : Peer
Metadata           : {
  }

MetadataMap        : {
  }

ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8f822ed6-42f3-4a26-911a-a4a6a87c0ef2
ServiceType        : MySnapIn
Version            : 1

Address            : http://MyServer.com:80/Citrix/MyContract/
    MyServiceEnvTestAPI/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : EnvironmentTest
Metadata           : {
  }

MetadataMap        : {
  }

ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : d2d40d9b-2a5d-4c5a-b9ca-a7f73cffe4f2
ServiceType        : MySnapIn
Version            : 1

Address            : http://MyServer.com:80/Citrix/MyContract/
    MyServiceAPI/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : InterService
Metadata           : {
  }

MetadataMap        : {
  }

ServiceAccount     : ENG\MyAccount$
```



```
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 5d428970-2ba1-4336-b8d0-f3aa961b8983
ServiceType        : MySnapIn
Version            : 1
```

EXAMPLE 2

Return all the service instances that are registered in the Configuration Service and are of type 'SDK'

.

```
C:\>Get-ConfigRegisteredServiceInstance -InterfaceType "SDK"

Address            : http://MyServer.com:80/Citrix/MyContract/v1
Binding            : wcf_HTTP_kerb
InterfaceType      : SDK
Metadata           : {
  }

MetadataMap        : {
  }

ServiceAccount     : ENG\MyAccount$
ServiceAccountSid  : S-1-5-21-1155438255-2213498043-2452000591-1104
ServiceGroupName   : MyService
ServiceGroupUid    : 2b990d5a-bba9-413b-aa08-e104e67f89bc
ServiceInstanceUid : 8dc38b5a-3fbb-457c-b326-6c41c94c18d5
ServiceType        : MySnapIn
Version            : 1
```

Parameters

-ServiceInstanceUid

The unique identifier for the service instance.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ServiceGroupUid

The unique identifier for the service group to which the service instance belongs.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ServiceGroupName

The name for the service group to which the service instance belongs.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ServiceType

The service type for the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Address

The connection address for the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Binding

The binding for the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Version

The service instance version.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ServiceAccountSid

The AD account SID for the computer account that the computer hosting the service instance is running as.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-InterfaceType

The interface type for the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Config_Filtering for details.

Type:	Int32

Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Config_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceInstance

This represents a service instance and has the following parameters:

- ServiceGroupUid <Guid>

  The unique identifier for the service group to which the service instance belongs.
- ServiceGroupName <string>

  The name of the service group to which the service instance belongs.
- ServiceInstanceUid <Guid>

  The unique identifier for the service instance.
- ServiceType <string>

  The type of the service group.
- Address <string>

  The contact address for the service instance.
- Binding <string>

  The binding to use for connections to the service instance.
- Version <int>

  The version of the service instance.
- ServiceAccount <string>

  The AD computer account for the computer that is providing the service instance.
- ServiceAccountSid <string>

  The AD computer account SID for the computer that is providing the service instance.
- InterfaceType <string>

  The interface type for the service instance.
- Metadata <Citrix.Configuration.Sdk.Metadata[]>

  The metadata for the service instance.

Notes

In the case of failure, the following errors can result.

Error Codes ————PartialData

Only a subset of the available data was returned.

CouldNotQueueryDatabase

The query required to get the database was not defined.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Register-ConfigServiceInstance
- Unregister-ConfigRegisteredServiceInstance
- Set-ConfigRegisteredServiceInstance

Get-ConfigService

March 11, 2024

Gets the service record entries for the Configuration Service.

Syntax

```
Get-ConfigService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
```



```
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the Configuration Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Configuration Service running in the current service group.

```
Get-ConfigService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.Service

The Get-ConfigServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin

Get-ConfigServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Configuration Service on the controller.

Syntax

```
Get-ConfigServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Enables updates to the Configuration Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Configuration Service.

```
1 Get-ConfigServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin

Get-ConfigServiceConfigurationData

March 11, 2024

Gets configuration data for the service.

Syntax

```
Get-ConfigServiceConfigurationData
    [-Name <String[]>]
    [-Value <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to determine the configuration parameters for the service

Examples

EXAMPLE 1

Get the Configuration data for the service.

```
Get-ConfigServiceConfigurationData

Name                : DeltaDiskDelete.timeDelay
Value               : 10
```

Parameters

-Name

The Configuration data item Name .

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Value

The Configuration data item value.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceConfigurationData

This object provides details of the configuration data and contains the following information:

Name <string>

The Name for the configuration item.

Value <string>

The value for the configuration item.

Notes

In the case of failure the following errors can be produced.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigServiceConfigurationData
- Remove-ConfigServiceConfigurationData

Get-ConfigServiceGroup

March 11, 2024

Gets the service groups that match the parameters supplied.

Syntax

```
Get-ConfigServiceGroup
   [-ServiceGroupUid <Guid>]
   [-ServiceGroupName <String>]
   [-ServiceType <String>]
   [-Metadata <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Use this cmdlet to retrieve existing service groups that match the parameters supplied. If no parameters are supplied, all the service groups are returned.

Examples

EXAMPLE 1

Return all the service groups that are registered in the Configuration Service.

```
C:\>Get-ConfigServiceGroup
```

EXAMPLE 2

Return all the service groups that are registered in the Configuration Service and are of type 'config' (i.e. the service groups for the Configuration Service).

```
C:\>Get-ConfigRegisteredServiceInstance -ServiceType "config"
```

Parameters

-ServiceGroupUid

The unique identifier for the service group.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ServiceGroupName

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceType

The service type for the service group.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Config_Filtering for details.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Config_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceGroup

This represents a service instance and has the following parameters; ServiceGroupUid <Guid>

The unique identifier for the service group.

ServiceGroupName <string>

The name of the service group.

ServiceType <string>

The type of the service group.

Metadata <Citrix.Configuration.Sdk.Metadata[]>

The metadata for the service group.

Notes

In the case of failure, the following errors can result.

Error Codes ————

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

PartialData

Only a subset of the available data was returned.

CouldNotQuueryDatabase

The Query required to get the database was not defined.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin

Get-ConfigServiceInstance

March 11, 2024

Gets the service instance entries for the Configuration Service.

Syntax

```
Get-ConfigServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Configuration Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Configuration Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-ConfigServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceInstance

The Get-ConfigServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Config.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Configuration.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigServiceStatus
- Reset-ConfigServiceGroupMembership

Get-ConfigServiceStatus

March 11, 2024

Gets the current state of the Configuration Service on the controller.

Syntax

```
Get-ConfigServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the Configuration Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Configuration Service.

```
Get-ConfigServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-ConfigServiceStatus command returns an object containing the status of the Configuration Service together with extra diagnostics information.

DBUnconfigured

The Configuration Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Configuration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Configuration Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Configuration is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Configuration is connected to a database that is valid, but it does not have the full functionality required so the Configuration cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Configuration Service currently in use is incompatible with the version of the Configuration Service schema on the database. Upgrade the Configuration Service to a more recent version.

DBOlderVersionThanService

The version of the Configuration Service schema on the database is incompatible with the version of the Configuration Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Configuration Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Configuration Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Configuration and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Configuration service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigDBConnection
- Test-ConfigDBConnection
- Get-ConfigDBConnection
- Get-ConfigDBSchema

Get-ConfigSite

March 11, 2024

Gets the site.

Syntax

```
Get-ConfigSite
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-ConfigSite cmdlet gets the site.

A XenDesktop installation has only a single site instance.

Examples

EXAMPLE 1

Gets the site.

```
Get-ConfigSite
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.Site

Provides information about the site

Related Links

- Set-ConfigSite
- Import-ConfigFeatureTable
- about_ConfigConfigurationSnapin

Get-ConfigZone

March 11, 2024

Gets zones configured for this site.

Syntax

```
Get-ConfigZone
   [-Uid <Guid>]
   [[-Name] <String>]
   [-ControllerName <String>]
   [-ControllerSid <String>]
   [-Description <String>]
   [-ExternalUid <Guid>]
   [-HasValidEdgeServers <Boolean>]
   [-IsHealthy <Boolean>]
   [-IsPrimary <Boolean>]
   [-Metadata <String>]
   [-TenantId <Guid>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
```



```
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves zones matching the specified criteria. If no parameters are specified all zones will be returned.

Examples

EXAMPLE 1

Gets the details of the zone with the specific name.

```
Get-ConfigZone -Name "London"
```

EXAMPLE 2

Lists all non-primary zones.

```
Get-ConfigZone -IsPrimary $false
```

Parameters

-Name

Gets zones with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Uid

Gets the zone with the specified UID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ControllerName

Gets the zone that contains a specified Controller identified by Name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ControllerSid

Gets the zone that contains a specified Controller identified by SID.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Description

Gets zones with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExternalUid

Specifies the ExternalUid of the zone. This is used for cloud sites to link with the Citrix Resource Location.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-HasValidEdgeServers

Gets zones with the specified value of the HasValidEdgeServers property.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsHealthy

Gets zones with the specified value of the IsHealthy property.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsPrimary

Gets zones with the specified value of the IsPrimary flag. If true, only the primary zone may be returned; if false, all other zones may be returned.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Gets the zones associated with the specified Tenant ID

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Config_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Config_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Configuration.Sdk.Zone

Get-ConfigZone returns an object for each matching zone.

Related Links

- New-ConfigZone
- Set-ConfigZone
- Rename-ConfigZone
- Remove-ConfigZone
- Set-ConfigSite
- Set-ConfigService

Import-ConfigFeatureTable

March 11, 2024

Sets the feature table of a site. This cmdlet can also be used to apply extra partial feature table files supplied by Citrix as needed to enable specific features. The XML feature table file is a signed file which Citrix provides for specific use cases on request.

Syntax

```
Import-ConfigFeatureTable
      [-Path] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Import-ConfigFeatureTable
      -Content <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Examples

EXAMPLE 1

Imports a feature table by specifying the path to a signed XML file provided by Citrix.

```
Import-ConfigFeatureTable -Path "XML_FeatureTable_File_Path\
   FeatureTableFilename.xml"
```

EXAMPLE 2

Imports the content of a signed XML feature table file provided by Citrix.

```
Import-ConfigFeatureTable -Content (Get-Content "
   XML_FeatureTable_File_Path\FeatureTableFilename.xml" | Out-String)
```


Parameters

-Path

The path to the file containing the feature table

Type:	String
Aliases:	PSPath
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Content

Set the site's feature table. (Can also update existing feature table by adding feature table fragments for specific extra features)

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Export-ConfigFeatureTable
- Get-ConfigSite
- Set-ConfigSite
- Get-ConfigProduct
- Get-ConfigProductEdition

- Get-ConfigProductFeature
- Get-ConfigProductVersion
- Get-ConfigLicensingModel
- about_ConfigConfigurationSnapin

New-ConfigEdgeServer

March 11, 2024

Creates a new edge server object

Syntax

```
New-ConfigEdgeServer
    [-Description <String>]
    -MachineAddress <String>
    [-Name] <String>
    -Sid <String>
    -Uuid <Guid>
    [-ZoneUid <Guid>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

New-ConfigEdgeServer cmdlet creates a new edge server object.

An edge server is a machine which is running an agent service that performs a number of duties; these agent services are largely stateless and communicate back to the Citrix Workspace Cloud components in the cloud.

Examples

EXAMPLE 1

This command creates an edge server

```
New-ConfigEdgeServer W2K12R2 -MachineAddress "EdgeSrv.address.com" -Sid
    21-3623811015-3361044348-30300820
```


Parameters

-Name

Specifies a name for the edge server.

Each edge server must have a unique name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-MachineAddress

The address used when contacted by local components such as VDAs

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Sid

The Security Identifier value in tenant AD domain

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Uuid

The uuid used when being contacted from cloud components such as controllers

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Optional value to specify a description for the edge server

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ZoneUid

Specifies the Zone affinity for the edge server

Type:	Guid
Position:	Named

Default value:	If no value is provided the edge server is placed in the Primary zone
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Configuration.Sdk.EdgeServer

The newly created edge server

Related Links

- Get-ConfigEdgeServer
- Set-ConfigEdgeServer
- Rename-ConfigEdgeServer
- Remove-ConfigEdgeServer

New-ConfigZone

March 11, 2024

Creates a new zone in the site.

Syntax

```
New-ConfigZone
   [-Description <String>]
   [-EnableHybridConnectivityForResourceLeases <Boolean>]
   [-EnableVdaConnectivityForResourceLeases <Boolean>]
   [-ExternalUid <Guid>]
   [-Name] <String>
   [-TenantId <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Creates a new zone in the site.

A Zone represents a subset of the site infrastructure residing in a particular geographical location.

By default any new zone will be marked as secondary. To specify the primary zone use the Set-ConfigSite cmdlet.

To associate controllers with a zone use the Set-ConfigService cmdlet.

Examples

EXAMPLE 1

Creates a new zone called 'London', marks it as the primary zone and moves a controller to that zone.

```
1 $zone = New-ConfigZone -Name "London" -Description "London branch
     office"
2 Set-ConfigSite -PrimaryZone $zone
3 Set-ConfigService -Service "S
     -1-5-21-3384143951-2794580461-1950386216-8227" -Zone $zone
```

Parameters

-Name

Specifies the name of the zone. Each zone in a site must have a unique name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Specifies the description of the zone.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-EnableHybridConnectivityForResourceLeases

Specifies the EnableHybridConnectivityForResourceLeases property associated with the zone. This enables both Gateway-Service and Direct launch for resource leases.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-EnableVdaConnectivityForResourceLeases

Specifies the EnableVdaConnectivityForResourceLeases property associated with the zone. This enables connectorless lease connection (ResolutionTypeVda) instead of lease connection with connectors.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ExternalUid

Specifies the ExternalUid of the zone. This is used for cloud sites to link with the Citrix Resource Location.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TenantId

Specifies the Tenant ID that the zone is associated with. This is used for cloud sites to associate zones with particular customers.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Configuration.Sdk.Zone

The newly created zone.

Related Links

- Get-ConfigZone
- Set-ConfigSite
- Set-ConfigService
- Set-ConfigZone
- Rename-ConfigZone
- Remove-ConfigZone

Register-ConfigServiceInstance

March 11, 2024

Allows the registration of a service instance.

Syntax

```
Register-ConfigServiceInstance
        [-LoggingId <Guid>]
        -ServiceInstance <ServiceInstance[]>
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Register-ConfigServiceInstance
        -ServiceGroupUid <Guid>
        -ServiceGroupName <String>
        -ServiceType <String>
        -Address <String>
        -Binding <String>
        -Version <Int32>
        -ServiceAccount <String>
        -InterfaceType <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Register-ConfigServiceInstance
        -ServiceGroupUid <Guid>
        -ServiceGroupName <String>
        -ServiceType <String>
        -Address <String>
        -Binding <String>
        -Version <Int32>
        -ServiceAccountSid <String>
        -InterfaceType <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Use this cmdlet to register service instance items in the Configuration Service. Service instances can be registered either by retrieving the data directly from other services or by manually entering the details into this command.

If the service group specified by the service instance already exists, the service is added to the service group, otherwise a new service group is created to hold the service instance.

Examples

EXAMPLE 1

Gets the service instances for the Configuration Service and registers them.

```
Get-ConfigServiceInstance | Register-ConfigServiceInstance
```

Parameters

-ServiceGroupUid

The Service Group Unique Identifier

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServiceGroupName

The name of the service group

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServiceType

The type of the service group

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Address

The address that is used to access the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Binding

The binding type that must be used to access the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Version

The version of the service instance.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServiceAccount

The AD computer account name (domain qualified) for the computer on which the service instance is hosted.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-InterfaceType

The type of interface that the service provides (i.e. SDK, InterService, Peer).

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServiceAccountSid

The AD computer account Sid for the computer on which the service instance is hosted.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServiceInstance

The service instances to register.

Type:	ServiceInstance[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the logging id of the high-level operation that this cmdlet is part of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.ServiceInstance

An object with the following parameters can be used to register a service instance:

- Address
- ServiceGroupUid
- ServiceGroupName
- ServiceType
- Binding
- Version
- InterfaceType

Outputs

Citrix.Configuration.Sdk.ServiceInstance

This represents a service instance and has the following parameters:

- ServiceGroupUid <Guid>

  The unique identifier for the service group to which the service instance belongs.

- ServiceGroupName <string>

  The name of the service group to which the service instance belongs.

- ServiceInstanceUid <Guid>

  The unique identifier for the service instance.

- ServiceType <string>

  The type of the service group.

- Address <string>

  The contact address for the service instance.

- Binding <string>

  The binding to use for connections to the service instance.

- Version <int>

  The version of the service instance.

- ServiceAccount <string>

  The AD computer account for the computer that is providing the service instance.

- ServiceAccountSid <string>

  The AD computer account SID for the computer that is providing the service instance.

- InterfaceType <string>

  The interface type for the service instance.

- Metadata <Citrix.Configuration.Sdk.Metadata[]>

  The metadata for the service instance.

Notes

In the case of failure, the following errors can result.

Error Codes ————ActiveDirectoryAccountResolutionFailed

The account name provided could not be found in Active Directory.

ServiceGroupWithSameUidExistsForDifferentServiceGroupNameOrSameUidExistsForDifferentServiceGroupNam

The service group name or service type do not match the service group found with the specified uid.

TypeAlreadyExists

A different service group with the same type is registered already in the Configuration Service.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Unregister-ConfigRegisteredServiceInstance
- Add-ConfigRegisteredServiceInstanceMetadata
- Set-ConfigRegisteredServiceInstanceMetadata
- Remove-ConfigRegisteredServiceInstanceMetadata
- Add-ConfigServiceGroupMetadata
- Set-ConfigServiceGroupMetadata
- Remove-ConfigServiceGroupMetadata

Remove-ConfigEdgeServer

March 11, 2024

Removes edge servers from the site

Syntax

```
Remove-ConfigEdgeServer
      [-InputObject] <EdgeServer[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServer
      [-Uid] <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServer
      [-Name] <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Remove edge servers from the site

Examples

EXAMPLE 1

Removes an edge server with the specified name

```
Remove-ConfigEdgeServer -Name W2K12R1
```

EXAMPLE 2

Removes all edge servers from the zone named Secondary

```
Get-ConfigEdgeServer -ZoneName Secondary | Remove-ConfigEdgeServer
```

Parameters

-InputObject

Specifies the edge server objects to delete

Type:	EdgeServer[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Specifies the UID of the edge server to delete

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the name of the edge server to delete

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.EdgeServer

You can pipe the edge servers to be deleted to Remove-ConfigEdgeServer.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-ConfigEdgeServer
- New-ConfigEdgeServer
- Set-ConfigEdgeServer
- Rename-ConfigEdgeServer

Remove-ConfigEdgeServerMetadata

March 11, 2024

Removes metadata from the given EdgeServer.

Syntax

```
Remove-ConfigEdgeServerMetadata
      [-EdgeServerUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServerMetadata
      [-EdgeServerUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServerMetadata
      [-EdgeServerName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServerMetadata
      [-EdgeServerName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigEdgeServerMetadata
      [-InputObject] <EdgeServer[]>
```



```
        -Name <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-ConfigEdgeServerMetadata
        [-InputObject] <EdgeServer[]>
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given EdgeServer.

Examples

EXAMPLE 1

Remove all metadata from all EdgeServer objects.

```
Get-ConfigEdgeServer | % {
  Remove-ConfigEdgeServerMetadata -Map $_.MetadataMap  }
```

Parameters

-EdgeServerUid

Id of the EdgeServer

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-EdgeServerName

Name of the EdgeServer

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	EdgeServer[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigEdgeServerMetadata

Remove-ConfigRegisteredServiceInstanceMetadata

March 11, 2024

Removes metadata from the given ServiceInstance.

Syntax

```
Remove-ConfigRegisteredServiceInstanceMetadata
      [-ServiceInstanceUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigRegisteredServiceInstanceMetadata
      [-ServiceInstanceUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ConfigRegisteredServiceInstanceMetadata
      [-InputObject] <ServiceInstance[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigRegisteredServiceInstanceMetadata
      [-InputObject] <ServiceInstance[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given ServiceInstance.

Examples

EXAMPLE 1

Remove all metadata from all ServiceInstance objects.

```
Get-ConfigServiceInstance | % {
  Remove-ConfigRegisteredServiceInstanceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceInstanceUid

Id of the ServiceInstance

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceInstance[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Add-ConfigRegisteredServiceInstanceMetadata
- Set-ConfigRegisteredServiceInstanceMetadata

Remove-ConfigServiceConfigurationData

March 11, 2024

Removes configuration data from the service.

Syntax

```
Remove-ConfigServiceConfigurationData
      [[-Name] <String>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove data from the Configuration Service configuration data.

Examples

EXAMPLE 1

Removes the configuration data item with name “MyName”.

```
Remove-ConfigServiceConfigurationData -Name "MyName"
```


Parameters

-Name

The name of the configuration data item to remove.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigServiceConfigurationData
- Get-ConfigServiceConfigurationData

Remove-ConfigServiceGroup

March 11, 2024

Removes service groups.

Syntax

```
Remove-ConfigServiceGroup
       [-ServiceGroupUid] <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Use this cmdlet to remove a service group and all the service instances that it contains.

Examples

EXAMPLE 1

Removes the service group (and all contained service instances) for the service group with a unique identifier of '31951f6f-7703-4abb-938a-2861d76ecfea'.


```
Remove-ConfigServiceGroup -ServiceGroupUid 31951f6f-7703-4abb-938a-2861
    d76ecfea
```

EXAMPLE 2

Removes all the service groups (and all contained service instances) for the XenDesktop deployment.

```
Get-ConfigServiceGroup | Remove-ConfigServiceGroup
```

Parameters

-ServiceGroupUid

The unique identifier for the service group.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the logging id of the high-level operation this cmdlet invocation is part of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes ————ServiceGroupObjectNotFound

The service group specified could not be located.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Register-ConfigServiceInstance
- Add-ConfigServiceGroupMetadata
- Set-ConfigServiceGroupMetadata
- Remove-ConfigServiceGroupMetadata

Remove-ConfigServiceGroupMetadata

March 11, 2024

Removes metadata from the given ServiceGroup.

Syntax

```
Remove-ConfigServiceGroupMetadata
      [-ServiceGroupUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigServiceGroupMetadata
      [-ServiceGroupUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigServiceGroupMetadata
      [-InputObject] <ServiceGroup[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ConfigServiceGroupMetadata
       [-InputObject] <ServiceGroup[]>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given ServiceGroup.

Examples

EXAMPLE 1

Remove all metadata from all ServiceGroup objects.

```
Get-ConfigServiceGroup | % {
  Remove-ConfigServiceGroupMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceGroupUid

Id of the ServiceGroup

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Add-ConfigServiceGroupMetadata
- Set-ConfigServiceGroupMetadata

Remove-ConfigServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-ConfigServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-ConfigService | % {
  Remove-ConfigServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigServiceMetadata

Remove-ConfigSiteMetadata

March 11, 2024

Removes metadata from the given Site.

Syntax

```
Remove-ConfigSiteMetadata
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigSiteMetadata
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Site.

Examples

EXAMPLE 1

Remove all metadata from all Site objects.

```
Get-ConfigSite | % {
  Remove-ConfigSiteMetadata -Map $_.MetadataMap  }
```

Parameters

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigSiteMetadata

Remove-ConfigZone

March 11, 2024

Removes a zone from the site.

Syntax

```
Remove-ConfigZone
      [-InputObject] <Zone[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ConfigZone
      [-Uid] <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ConfigZone
       [-Name] <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This cmdlet removes zones from a site.

You cannot remove a zone that is marked as primary or has associated controllers.

Examples

EXAMPLE 1

Remove the zone specified by name. This fails if one of the zones has controllers associated.

```
Remove-ConfigZone -Name 'Sydney'
```

EXAMPLE 2

Move all controllers to the primary zone. Remove all non-primary zones.

```
Get-ConfigService | Set-ConfigService -Zone (Get-ConfigZone -IsPrimary
   $true)
Get-ConfigZone -IsPrimary $false | Remove-ConfigZone
```

Parameters

-InputObject

Specifies the zone to remove (by zone object).

Type:	Zone[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Uid

Specifies the zone to remove (by Uid).

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the zone to remove (by Name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.Zone

You can pipe the zones to be deleted into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-ConfigZone
- New-ConfigZone
- Set-ConfigZone
- Rename-ConfigZone
- Set-ConfigSite

- Set-ConfigService

Remove-ConfigZoneMetadata

March 11, 2024

Removes metadata from the given Zone.

Syntax

```
Remove-ConfigZoneMetadata
       [-ZoneUid] <Guid>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ConfigZoneMetadata
       [-ZoneUid] <Guid>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ConfigZoneMetadata
       [-ZoneName] <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ConfigZoneMetadata
       [-ZoneName] <String>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ConfigZoneMetadata
       [-InputObject] <Zone[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ConfigZoneMetadata
```



```
        [-InputObject] <Zone[]>
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Zone.

Examples

EXAMPLE 1

Remove all metadata from all Zone objects.

```
Get-ConfigZone | % {
  Remove-ConfigZoneMetadata -Map $_.MetadataMap  }
```

Parameters

-ZoneUid

Id of the Zone

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ZoneName

Name of the Zone

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Zone[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Set-ConfigZoneMetadata

Rename-ConfigEdgeServer

March 11, 2024

Renames an edge server

Syntax

```
Rename-ConfigEdgeServer
      [-InputObject] <EdgeServer>
      [-NewName] <String>
      [-PassThru]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-ConfigEdgeServer
      [-Uid] <Guid>
      [-NewName] <String>
      [-PassThru]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-ConfigEdgeServer
      [-Name] <String>
      [-NewName] <String>
```



```
        [-PassThru]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

The Rename-ConfigEdgeServer cmdlet changes the name of an edge server.

All edge servers in a site must have a unique name.

The following special characters are not allowed in the name: \ / ; : # . * ? = < > | [ ] ( ) “ ‘

Examples

EXAMPLE 1

Renames the edge server with the name “Old name”to “New Name”

```
Rename-ConfigEdgeServer -Name "Old name" -NewName "New Name"
```

EXAMPLE 2

Renames the edge server with the name “Old name”to “New Name”

```
 Get-ConfigEdgeServer "Old name" | Rename-ConfigEdgeServer -NewName "
    New name"
```

Parameters

-InputObject

Specifies the edge server to rename

Type:	EdgeServer
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Specifies the UID of the edge server to rename

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the name of the edge server to rename

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the edge server

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.EdgeServer

You can pipe edge servers to Rename-ConfigEdgeServer.

Outputs

None or Citrix.Configuration.Sdk.EdgeServer

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Configuration.Sdk.EdgeServer object.

Related Links

- New-ConfigEdgeServer
- Set-ConfigEdgeServer
- Get-ConfigEdgeServer
- Remove-ConfigEdgeServer

Rename-ConfigZone

March 11, 2024

Rename a zone.

Syntax

```
Rename-ConfigZone
       [-InputObject] <Zone>
       [-NewName] <String>
       [-PassThru]
```



```
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Rename-ConfigZone
        [-Uid] <Guid>
        [-NewName] <String>
        [-PassThru]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Rename-ConfigZone
        [-Name] <String>
        [-NewName] <String>
        [-PassThru]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This cmdlet renames a zone.

All zone names must be unique.

Examples

EXAMPLE 1

Renames the ‘New York’zone to ‘Manhattan’.

```
Rename-ConfigZone -Name 'New York' -NewName 'Manhattan'
```

Parameters

-InputObject

Specifies the zone to rename (by zone object).

Type:	Zone
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Specifies the zone to rename (by Uid).

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the zone to rename (by name).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the zone.

Type:	String
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.Zone

You can pipe the zone to be renamed into this command.

Outputs

None or Citrix.Configuration.Sdk.Zone

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it returns the zone with new name.

Related Links

- New-ConfigZone
- Set-ConfigZone
- Get-ConfigZone
- Remove-ConfigZone
- Set-ConfigSite
- Set-ConfigService

Reset-ConfigEdgeServerList

March 11, 2024

Requests a refresh of the list of Edge Servers.

Syntax

```
Reset-ConfigEdgeServerList
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

This cmdlet can be used to request that the service refreshes its internal list of Edge Servers.

This will effectively synchronize the list of Edge Servers with the one of the Central Configuration Service.

Examples

EXAMPLE 1

Refresh the internal list of Edge Servers.

```
Reset-ConfigEdgeServerList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin

Reset-ConfigEnabledFeatureList

March 11, 2024

Refreshes the Configuration service's list of enabled features.

Syntax

```
Reset-ConfigEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Configuration Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. However, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Configuration service instance's list of enabled features.

```
Reset-ConfigEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-ConfigServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Configuration Service.

Syntax

```
Reset-ConfigServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Configuration Service access permissions and configuration service locations. The Reset-ConfigServiceGroupMembership command must be run on at least one instance of the service type (Config) after installation and registration with the configuration service. Without this

operation, the Configuration services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-ConfigServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
1 Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    ConfigServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
1 Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-ConfigServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-ConfigServiceGroupMembership command.

Outputs

Citrix.Configuration.Sdk.ServiceInstance

Reset-ConfigServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Configuration Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigServiceInstance
- Get-ConfigServiceStatus

Set-ConfigConnectorAppliance

March 11, 2024

Changes the properties of a connector appliance

Syntax

```
Set-ConfigConnectorAppliance
   [-InputObject] <ConnectorAppliance[]>
   [-ConnectorType <String>]
   [-Description <String>]
   [-IsHealthy <Boolean>]
   [-MachineAddress <String>]
   [-Uuid <Guid>]
   [-ZoneUid <Guid>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigConnectorAppliance
   [-Uid] <Guid[]>
   [-ConnectorType <String>]
   [-Description <String>]
   [-IsHealthy <Boolean>]
   [-MachineAddress <String>]
   [-Uuid <Guid>]
   [-ZoneUid <Guid>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigConnectorAppliance
   [-Name] <String[]>
   [-ConnectorType <String>]
   [-Description <String>]
   [-IsHealthy <Boolean>]
   [-MachineAddress <String>]
```



```
    [-Uuid <Guid>]
    [-ZoneUid <Guid>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-ConfigConnectorAppliance cmdlet sets properties of a connector appliance

Examples

Parameters

-InputObject

Specifies the connector appliance objects to modify.

Type:	ConnectorAppliance[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Identifies the connector appliance to modify by its UID property.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Identifies the connector appliance to modify by name.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ConnectorType

Sets the connector types

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Supplies the new value of the Description property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsHealthy

Supplies the new value of the IsHealthy property.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineAddress

Supplies the new value of the MachineAddress property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Uuid

Supplies the new value of the Uuid property.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Allows moving the connector appliance to a new zone

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.ConnectorAppliance

You can pipe the connector appliances to be updated into this command.

Outputs

None or Citrix.Configuration.Sdk.ConnectorAppliance

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Configuration.Sdk.ConnectorAppliance object.

Related Links

Set-ConfigDBConnection

March 11, 2024

Configures a database connection for the Configuration Service.

Syntax

```
Set-ConfigDBConnection
    [-DBConnection] <String>
    [-Force]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Configuration Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-ConfigDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```


EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
1  Set-ConfigDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Configuration Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-ConfigDBConnection cmdlet returns an object describing the status of the Configuration Service together with extra diagnostics information. Possible values are:

- OK:

The Configuration Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Configuration Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Configuration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Configuration Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Configuration Service currently in use is newer than, and incompatible with, the version of the Configuration Service schema on the database. Upgrade the Configuration Service to a more recent version.

- DBOlderVersionThanService:

The version of the Configuration Service schema on the database is newer than, and incompatible with, the version of the Configuration Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Configuration Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Configuration Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Configuration Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigServiceStatus
- Get-ConfigDBConnection
- Test-ConfigDBConnection

Set-ConfigDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Configuration Service.

Syntax

```
Set-ConfigDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Configuration Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Configuration Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-ConfigDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-ConfigDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-ConfigDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-ConfigDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-ConfigDBSchema
- Set-ConfigDBConnection
- Get-Credential

Set-ConfigEdgeServer

March 11, 2024

Changes the properties of an edge server

Syntax

```
Set-ConfigEdgeServer
    [-InputObject] <EdgeServer[]>
    [-Description <String>]
    [-IsHealthy <Boolean>]
    [-MachineAddress <String>]
    [-Sid <String>]
    [-Uuid <Guid>]
    [-ZoneUid <Guid>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigEdgeServer
    [-Uid] <Guid[]>
    [-Description <String>]
    [-IsHealthy <Boolean>]
    [-MachineAddress <String>]
    [-Sid <String>]
    [-Uuid <Guid>]
    [-ZoneUid <Guid>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigEdgeServer
    [-Name] <String[]>
    [-Description <String>]
    [-IsHealthy <Boolean>]
    [-MachineAddress <String>]
    [-Sid <String>]
    [-Uuid <Guid>]
    [-ZoneUid <Guid>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

The Set-ConfigEdgeServer cmdlet sets properties of an edge server or a set of edge servers. The edge server can be specified by name or by one or more edge server instances can be passed to the command either by piping or by using the -InputObject parameter

Examples

EXAMPLE 1

Associates the edge server 'EdgeSrv1'to a zone specified by its UID

```
Set-ConfigEdgeServer EdgeSrv1 -ZoneUid d8fe27fa-f33c-47ee-b6b5-80241
    f536164
```

Parameters

-InputObject

Specifies the edge server objects to modify.

Type:	EdgeServer[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Identifies the edge server to modify by its UID property.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Identifies the edge server to modify by name.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Supplies the new value of the Description property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsHealthy

Supplies the new value of the IsHealthy property.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineAddress

Supplies the new value of the MachineAddress property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Supplies the new value of the SID property.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Uuid

Supplies the new value of the Uuid property.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneUid

Allows moving the edge server to a new zone

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.EdgeServer

You can pipe the edge servers to be updated into this command.

Outputs

None or Citrix.Configuration.Sdk.EdgeServer

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.Configuration.Sdk.EdgeServer object.

Related Links

- New-ConfigEdgeServer
- Get-ConfigEdgeServer
- Rename-ConfigEdgeServer
- Remove-ConfigEdgeServer

Set-ConfigEdgeServerMetadata

March 11, 2024

Adds or updates metadata on the given EdgeServer.

Syntax

```
Set-ConfigEdgeServerMetadata
   [-EdgeServerUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigEdgeServerMetadata
   [-EdgeServerUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigEdgeServerMetadata
   [-EdgeServerName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigEdgeServerMetadata
   [-EdgeServerName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ConfigEdgeServerMetadata
   [-InputObject] <EdgeServer[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigEdgeServerMetadata
   [-InputObject] <EdgeServer[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given EdgeServer objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the EdgeServer with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ConfigEdgeServerMetadata -EdgeServerUid 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-EdgeServerUid

Id of the EdgeServer

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-EdgeServerName

Name of the EdgeServer

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	EdgeServer[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the EdgeServer specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigEdgeServerMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Remove-ConfigEdgeServerMetadata

Set-ConfigRegisteredServiceInstance

March 11, 2024

Updates a service instance.

Syntax

```
Set-ConfigRegisteredServiceInstance
   -ServiceInstanceUid <Guid>
   -Address <String>
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this cmdlet to change the address property of an existing service instance that is registered in the Configuration Service.

Examples

EXAMPLE 1

Update the service instance with the unique identifier of '9805f39d-99eb-44f0-8f63-9d8e3f1228e0'to use the new address.

```
Set-ConfigRegisteredServiceInstance -ServiceInstanceUid "9805f39d-99eb
   -44f0-8f63-9d8e3f1228e0" -Address "http://myServer.com/Citrix/
   sdkHostingUnitService"
```


Parameters

-ServiceInstanceUid

The unique identifier for the service instance to be updated.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Address

The new address for the service instance.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Defines whether or not the command returns a result showing the new state of the updated service instance.

Type:	SwitchParameter
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LoggingId

Specifies the logging id of the high-level operation this cmdlet invocation is part of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceInstance

This represents a service instance and has the following parameters:

- ServiceGroupUid <Guid>

  The unique identifier for the service group to which the service instance belongs.

- ServiceGroupName <string>

  The name of the service group to which the service instance belongs.

- ServiceInstanceUid <Guid>

  The unique identifier for the service instance.

- ServiceType <string>

  The type of the service group.

- Address <string>

  The contact address for the service instance.

- Binding <string>

  The binding to use for connections to the service instance.

- Version <int>

  The version of the service instance.

- ServiceAccount <string>

  The AD computer account for the computer that is providing the service instance.

- ServiceAccountSid <string>

  The AD computer account SID for the computer that is providing the service instance.

- InterfaceType <string>

  The interface type for the service instance.

- Metadata <Citrix.Configuration.Sdk.Metadata[]>

  The metadata for the service instance.

Notes

In the case of failure, the following errors can result.

Error Codes ————ObjectToUpdateDoesNotExist

The service instance specified could not be located.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigRegisteredServiceInstance
- Register-ConfigServiceInstance
- Unregister-ConfigRegisteredServiceInstance

Set-ConfigRegisteredServiceInstanceMetadata

March 11, 2024

Adds or updates metadata on the given ServiceInstance.

Syntax

```
Set-ConfigRegisteredServiceInstanceMetadata
   [-ServiceInstanceUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigRegisteredServiceInstanceMetadata
   [-ServiceInstanceUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigRegisteredServiceInstanceMetadata
    [-InputObject] <ServiceInstance[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigRegisteredServiceInstanceMetadata
    [-InputObject] <ServiceInstance[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given ServiceInstance objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the ServiceInstance with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-ConfigRegisteredServiceInstanceMetadata -ServiceInstanceUid 4
    CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-ServiceInstanceUid

Id of the ServiceInstance

Type:	Guid
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceInstance[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the ServiceInstance specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigRegisteredServiceInstanceMetadata returns a dictionary containing the new (name, value)-pairs.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Add-ConfigRegisteredServiceInstanceMetadata
- Remove-ConfigRegisteredServiceInstanceMetadata

Set-ConfigService

March 11, 2024

Associate a Service with a Zone.

Syntax

```
Set-ConfigService
   [-InputObject] <Service[]>
   [-Zone] <Zone>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigService
   [-Uid] <Int32[]>
   [-Zone] <Zone>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigService
   [-MachineName] <String[]>
   [-Zone] <Zone>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This cmdlet is used to change the association between Services (referred also as Controllers) and Zones. A Service can be associated with only one Zone. By default, all services are assigned to the primary Zone.

Examples

EXAMPLE 1

Associate a Service identified by SID with a Zone identified by name.

```
Set-ConfigService -InputObject "S
   -1-5-21-3384143951-2794580461-1950386216-8227" -Zone "London"
```


Parameters

-InputObject

The service to associate with a Zone.

The service can be referenced by Service object, Machine Name, Machine DNS Name, SID or UID.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Allows specifiying Services by the Service Uid

Type:	Int32[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MachineName

Allows specifiying Services by the Machine Name

Type:	String[]
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Zone

Specifies the target Zone.

Zone can be referenced by Zone object, Name or UID.

Type:	Zone
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Fma.Sdk.CommonCmdlets.Service

You can pipe the Services to be associated with the Zone.

Outputs

None

By default, this cmdlet returns no output.

Notes

For historical reasons Service is also referred as Controller in FMA services.

Related Links

- Get-ConfigService
- Get-ConfigZone
- New-ConfigZone
- Set-ConfigSite

Set-ConfigServiceConfigurationData

March 11, 2024

Sets the value for the given key in the service configuration data.

Syntax

```
Set-ConfigServiceConfigurationData
    [-Name] <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored for the Configuration Service.

Examples

EXAMPLE 1

Set data with a name of ‘customProperty1’and value of ‘value2’to the service configuration.

```
Set-ConfigServiceConfigurationData -Name "customProperty1" -Value "
    value2"

Name                           Value
--------                       -----
customProperty1                value2
```

Parameters

-Name

Specifies the key name of the metadata to be added. The key must be unique.

The Name cannot contain any of the following characters \/;:#.*?=<>|[]()’””

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the name. If the name already exists, its value is updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Configuration.Sdk.ServiceConfigurationData

Set-ConfigServiceConfigurationData returns an object containing the new definition of the configuration.

Name <string>

Specifies the name for the item of data.

Value <string>

Specifies the value of the data.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Remove-ConfigServiceConfigurationData
- Get-ConfigServiceConfigurationData

Set-ConfigServiceGroupMetadata

March 11, 2024

Adds or updates metadata on the given ServiceGroup.

Syntax

```
Set-ConfigServiceGroupMetadata
   [-ServiceGroupUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigServiceGroupMetadata
   [-ServiceGroupUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ConfigServiceGroupMetadata
   [-InputObject] <ServiceGroup[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigServiceGroupMetadata
   [-InputObject] <ServiceGroup[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given ServiceGroup objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the ServiceGroup with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-ConfigServiceGroupMetadata -ServiceGroupUid 4CECC26E-48E1-423F-A1F0
   -2A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-ServiceGroupUid

Id of the ServiceGroup

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceGroup[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the ServiceGroup specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigServiceGroupMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Add-ConfigServiceGroupMetadata
- Remove-ConfigServiceGroupMetadata

Set-ConfigServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-ConfigServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ConfigServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                      Value
```



```
---                                  -----
property                             value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Remove-ConfigServiceMetadata

Set-ConfigSite

March 11, 2024

Changes the overall settings of the site.

Syntax

```
Set-ConfigSite
    [-SiteName <String>]
    [-ProductCode <String>]
    [-ProductEdition <String>]
    [-ProductVersion <String>]
    [-LicensingModel <String>]
    [-LicenseServerName <String>]
    [-LicenseServerPort <Int32>]
    [-LicenseServerUri <Uri>]
    [-PrimaryZone <Zone>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-ConfigSite cmdlet modifies properties of the site.

The site is a top-level, logical representation of the XenDesktop site, from the perspective of the configuration services running within the site.

A XenDesktop installation has only a single site instance.

Modifications to the product code, product edition, product version and licensing model properties are successful only if their values are consistent with the feature table. Use the Get-ConfigProduct, Get-ConfigProductEdition, Get-ConfigProductVersion and Get-ConfigLicensingModel cmdlets to determine consistent values.

To configure the site, first import the feature table using the Import-ConfigFeatureTable cmdlet.

Examples

EXAMPLE 1

Specifies the use of a Platinum edition license. A suitable license must be available on the site's license server.


```
Set-ConfigSite -ProductEdition PLT
```

Parameters

-SiteName

Changes the name of the site.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductCode

Changes the product code.

The Get-ConfigProduct cmdlet returns a list of supported values.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductEdition

Changes the license edition. A license matching the specified edition must be available within the site's license server.

The Get-ConfigProductEdition returns a list of supported values.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductVersion

Changes the product version.

The Get-ConfigProductVersion returns a list of supported values.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicensingModel

Changes the license model. A license matching the specified model must be available within the site's license server.

The Get-ConfigLicensingModel returns a list of supported values.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseServerName

Changes the machine used by the brokering services to obtain licenses for desktop and application session brokering. The specified machine must be running a Citrix license server and have suitable licenses installed.

The license server machine can be specified by its DNS name ('machine.domain') or its numeric IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseServerPort

Changes the port number on the license server machine used by the brokering services to contact the Citrix license server.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseServerUri

Changes the Uri of the web server for licensing. The hostname component of this Uri must match the Site's LicenseServerName property.

Type:	Uri
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrimaryZone

Changes the primary zone for the site.

Primary zone can be specified by Uid, Name or by passing a Zone object.

Type:	Zone
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Site

Related Links

- Export-ConfigFeatureTable
- Get-ConfigSite
- Get-ConfigProduct
- Get-ConfigProductEdition
- Get-ConfigProductFeature
- Get-ConfigProductVersion
- Get-ConfigLicensingModel
- about_ConfigConfigurationSnapin

Set-ConfigSiteMetadata

March 11, 2024

Adds or updates metadata on the Site.

Syntax

```
Set-ConfigSiteMetadata
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigSiteMetadata
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against the Site.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Site.

```
Set-ConfigSiteMetadata -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Site specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()’”

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigSiteMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Remove-ConfigSiteMetadata

Set-ConfigZone

March 11, 2024

Set the description of the zone.

Syntax

```
Set-ConfigZone
   [-NewUid <Guid>]
   [-InputObject] <Zone[]>
   [-Description <String>]
   [-EnableHybridConnectivityForResourceLeases <Boolean>]
   [-EnableVdaConnectivityForResourceLeases <Boolean>]
   [-ExternalUid <Guid>]
   [-TenantId <Guid>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigZone
   [-NewUid <Guid>]
   [-Uid] <Guid[]>
   [-Description <String>]
   [-EnableHybridConnectivityForResourceLeases <Boolean>]
   [-EnableVdaConnectivityForResourceLeases <Boolean>]
   [-ExternalUid <Guid>]
   [-TenantId <Guid>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ConfigZone
   [-NewUid <Guid>]
   [-Name] <String[]>
   [-Description <String>]
   [-EnableHybridConnectivityForResourceLeases <Boolean>]
   [-EnableVdaConnectivityForResourceLeases <Boolean>]
   [-ExternalUid <Guid>]
   [-TenantId <Guid>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

This cmdlet allows you to change the description of a zone.

Controllers can be moved between zones using Set-ConfigService cmdlet. To mark zone as primary use Set-ConfigSite cmdlet.

To update the metadata associated with a zone, use the Set-ConfigZoneMetadata and Remove-ConfigZoneMetadata cmdlets.

To change the name of a zone use Rename-ConfigZone cmdlet.

Examples

EXAMPLE 1

Change the description of the 'Sydney'zone.

```
Set-ConfigZone -Name 'Sydney' -Description 'Sydney branch office'
```

Parameters

-InputObject

Specifies the zone to update (by zone object).

Type:	Zone[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Specifies the zone to update (by Uid).

Type:	Guid[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the zone to update (by name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-NewUid

Specifies the new uid of the Zone object.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Supplies the new description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableHybridConnectivityForResourceLeases

Supplies the EnableHybridConnectivityForResourceLeases property of the zone

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableVdaConnectivityForResourceLeases

Supplies the EnableVdaConnectivityForResourceLeases property of the zone

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExternalUid

Supplies the external Uid of the zone

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Supplies the Tenant ID of the zone

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.Zone

You can pipe the zones to be updated into this command.

Outputs

None or Citrix.Configuration.Sdk.Zone

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Zone object.

Related Links

- New-ConfigZone
- Get-ConfigZone
- Rename-ConfigZone
- Remove-ConfigZone
- Set-ConfigSite
- Set-ConfigService

Set-ConfigZoneMetadata

March 11, 2024

Adds or updates metadata on the given Zone.

Syntax

```
Set-ConfigZoneMetadata
    [-ZoneUid] <Guid>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigZoneMetadata
    [-ZoneUid] <Guid>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigZoneMetadata
    [-ZoneName] <String>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigZoneMetadata
    [-ZoneName] <String>
    -Map <PSObject>
    [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigZoneMetadata
    [-InputObject] <Zone[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ConfigZoneMetadata
    [-InputObject] <Zone[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Zone objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Zone with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ConfigZoneMetadata -ZoneUid 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -
    Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-ZoneUid

Id of the Zone

Type:	Guid
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ZoneName

Name of the Zone

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Zone[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Zone specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ConfigZoneMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Remove-ConfigZoneMetadata

Test-ConfigDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Configuration Service.

Syntax

```
Test-ConfigDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ConfigDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ConfigDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Configuration Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Configuration SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-ConfigDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Configuration Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login

and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-ConfigDBConnection cmdlet returns an object describing the status of the selected Configuration Service instance that would result if the connection string were used with the Set-

ConfigDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-ConfigDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Configuration Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Configuration Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Configuration Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Configuration Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Configuration Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Configuration Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Get-ConfigServiceStatus
- Get-ConfigDBConnection
- Set-ConfigDBConnection

Test-ConfigServiceInstanceAvailability

March 11, 2024

Tests whether the supplied service instances are responding to requests.

Syntax

```
Test-ConfigServiceInstanceAvailability
    [-ServiceInstance] <ServiceInstance[]>
    [-MaxDelaySeconds <Int32>]
    [-ForceWaitForOneOfEachType]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Examples

EXAMPLE 1

Test all the service instances that are registered in the Configuration Service, returning when one of each type is responding.

```
C:\>Get-ConfigRegisteredServiceInstance | Test-
   ConfigServiceInstanceAvailability -ForceWaitForOneOfEachType
```

EXAMPLE 2

Test each of the given services, allowing a 5 second time-out.

```
C:\>Test-ConfigServiceInstanceAvailability -ServiceInstance $services -
   MaxDelaySeconds 5
```

Parameters

-ServiceInstance

The service instances to test.

Type:	ServiceInstance[]
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-MaxDelaySeconds

The timeout period to wait before concluding that services are unresponsive.

Type:	Int32
Position:	Named
Default value:	Infinite
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ForceWaitForOneOfEachType

If at least one of each type of service responds, finish immediately.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

This represents a service instance and has the following parameters:

- ServiceGroupUid <Guid>

  The unique identifer for the service group to which the service instance belongs.
- ServiceGroupName <string>

  The name of the service group that the service instance is part of.
- ServiceInstanceUid <Guid>

  The unique identifier for the service instance.
- ServiceType <string>

  The type of the service group.
- Address <string>

  The contact address for the service instance.
- Binding <string>

  The binding to use for connections to the service instance.
- Version <int>

  The version of the service instance.
- ServiceAccount <string>

  The AD computer account for the computer that is providing the service instance.

- ServiceAccountSid <string>

  The AD computer account SID for the computer that is providing the service instance.

- InterfaceType <string>

  The interface type for the service instance.

- Metadata <Citrix.Configuration.Sdk.Metadata[]>

  The metadata for the service instance.

- Status <Citrix.Configuration.Sdk.Commands.Availability>

  An enumeration value indicating whether the service is Responding, NotResponding, Unknown, or BadBindingType.

- ResponseTime <System.TimeSpan>

  The interval elapsed between hailing the service and getting a definite response

Notes

The Availability Status Codes are

- Responding: Got a positive response
- NotResponding: Got a response, but it was negative or the connection was refused
- Unknown: Did not respond in time / timed-out
- BadBindingType: Binding parameter in ServiceInstance is not wcf_HTTP_kerb

Related Links

- about_ConfigConfigurationSnapin
- Register-ConfigServiceInstance
- Unregister-ConfigRegisteredServiceInstance
- Get-ConfigRegisteredServiceInstance

Unregister-ConfigRegisteredServiceInstance

March 11, 2024

Removes a service instance from the Configuration Service registry.

Syntax

```
Unregister-ConfigRegisteredServiceInstance
           [-ServiceInstanceUid] <Guid>
           [-LoggingId <Guid>]
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```

Description

Use this cmdlet to remove a service instance from the Configuration Service registry. This does not remove any service groups (if all service instances for a Service Group are removed, an empty service group remains and must be removed using the Remove-ConfigServiceGroup command).

Examples

EXAMPLE 1

Unregisters all the service instances that are for a service type of 'Config'from the Configuration Service instance register.

```
Get-ConfigRegisteredServiceInstance -ServiceType "Config" | Unregister-
    ConfigRegisteredServiceInstance
```

Parameters

-ServiceInstanceUid

The unique identifier for the service instance to be removed.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the logging id of the high-level operation this cmdlet invocation is part of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Configuration.Sdk.ServiceInstance

An object with a parameter called 'ServiceInstanceUid'can be used to unregister service instances.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes ————ServiceInstanceObjectNotFound

The service instance specified could not be located.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ConfigConfigurationSnapin
- Register-ConfigServiceInstance

about_LogConfigurationLoggingSnapIn

March 11, 2024

Topic

about_LogConfigurationLoggingSnapin

Short Description

The Configuration Logging Service PowerShell snap-in provides administrative functions for the Configuration Logging Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Log'.

Long Description

The Configuration Logging Service PowerShell snap-in enables both local and remote administration of the Configuration Logging Service.

The Configuration Logging Service logs configuration changes or administrator requested state changes made to the site. Configuration Logging can be configured, site wide, to be mandatory or optional. If mandatory logging is selected, then any attempts to change site configuration or state when the logging mechanism is unavailable are denied.

The Configuration Logging Service stores information about the logged changes in a database which can be configured to be separate from the site database.

The snap-in provides storage and configuration of these entities:

Site

The Configuration Logging Site object holds global settings which control the behaviour of the Configuration Logging Service. The site object can be configured by the Set-LogSite cmdlet. The properties of the site object are returned by the Get-LogSite cmdlet.

High Level Operations

A high level operation object represents a logged configuration change performed from Desktop Studio, Desktop Director or a PowerShell Script.

The XenDesktop consoles log high level operations when:

1. Executing operations which performs configuration changes.
2. Executing operations which performs administration related activities which may affect site configuration.

PowerShell scripts which carry out customized configuration changes can also log high level operations via cmdlets Start-LogHighLevelOperation and Stop-LogHighLevelOperation.

Low Level Operations

A low level operation object represents a logged configuration change performed by a service. One or more low level operation objects are used to record the actions performed by a services in order to fulfil a high level operation initiated from the consoles, or from PowerShell scripts.

Low level operations in the system are returned by cmdlet Get-LogLowLevelOperation.

Operation Details

A low level operation performed by a service can affect a number of individual objects, or a number of properties on an object. An operation detail log records each individual change to an object. This includes the creation and deletion of the object, as well as changes to individual properties of the object.

One or more operation detail objects are used to record specific changes to each object that is affected by a low level service operation.

Operation details are included in the data returned from the Get-LogLowLevelOperation cmdlet.

High Level Operations, Low Level Operations and Operation Details are arranged in a hierarchy. A High Level Operation can have multiple Low Level Operations, and each Low Level Operation can have multiple Operation Details.

Setting Up A Separate Logging Database

After creating the database on the database server, the logging database can be setup and configured for use by:

1. Generating the database schema, and applying it the logging database
2. Configuring the configuration logging service to use the new logging database.

The logging database schema can be generated from the Get-LogDBSchema cmdlet, as illustrated below:

```
1 Get-LogDBSchema -DatabaseName "loggingDB"
```



```
                -ServiceGroupName "service group name"
                -ScriptType Database
                -LocalDatabase:$LocalDB
                -DataStore Logging
```

The configuration logging service can be configured to use the logging database with the Set-LogDBConnection cmdlet, as illustrated below:

```
Set-LogDBConnection -DataStore Logging -DBConnection $null
Set-LogDBConnection -DataStore Logging -DBConnection "new logging db
    connection string"
```

Configuration Logging Site Settings

On the Site object:

- The 'State'setting allows configuration logging to be disabled, enabled, or made mandatory.
- The 'Locale'setting specifies the language in which configuration logging data text will be stored.

See Get-LogSite cmdlet help for further information on these settings.

This locale setting applies to the text description that is associated with each log , e.g. 'Create Catalog'. It doesn't apply to other textual information in the log like the names of parameters passed to operations, e.g. 'CatalogName'.

This localisation is applied when the data is logged, and not when the logs are viewed later. For example, logs which are created in English will be displayed in English on an end user system which may be configured with a different locale.

about_LogConfigurationLoggingSnapIn

March 11, 2024

Topic

about_LogConfigurationLoggingSnapin

Short Description

The Configuration Logging Service PowerShell snap-in provides administrative functions for the Configuration Logging Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Log'.

Long Description

The Configuration Logging Service PowerShell snap-in enables both local and remote administration of the Configuration Logging Service.

The Configuration Logging Service logs configuration changes or administrator requested state changes made to the site. Configuration Logging can be configured, site wide, to be mandatory or optional. If mandatory logging is selected, then any attempts to change site configuration or state when the logging mechanism is unavailable are denied.

The Configuration Logging Service stores information about the logged changes in a database which can be configured to be separate from the site database.

The snap-in provides storage and configuration of these entities:

Site

The Configuration Logging Site object holds global settings which control the behaviour of the Configuration Logging Service. The site object can be configured by the Set-LogSite cmdlet. The properties of the site object are returned by the Get-LogSite cmdlet.

High Level Operations

A high level operation object represents a logged configuration change performed from Desktop Studio, Desktop Director or a PowerShell Script.

The XenDesktop consoles log high level operations when:

1. Executing operations which performs configuration changes.
2. Executing operations which performs administration related activities which may affect site configuration.

PowerShell scripts which carry out customized configuration changes can also log high level operations via cmdlets Start-LogHighLevelOperation and Stop-LogHighLevelOperation.

Low Level Operations

A low level operation object represents a logged configuration change performed by a service. One or more low level operation objects are used to record the actions performed by a services in order to fulfil a high level operation initiated from the consoles, or from PowerShell scripts.

Low level operations in the system are returned by cmdlet Get-LogLowLevelOperation.

Operation Details

A low level operation performed by a service can affect a number of individual objects, or a number of properties on an object. An operation detail log records each individual change to an object. This includes the creation and deletion of the object, as well as changes to individual properties of the object.

One or more operation detail objects are used to record specific changes to each object that is affected by a low level service operation.

Operation details are included in the data returned from the Get-LogLowLevelOperation cmdlet.

High Level Operations, Low Level Operations and Operation Details are arranged in a hierarchy. A High Level Operation can have multiple Low Level Operations, and each Low Level Operation can have multiple Operation Details.

Setting Up A Separate Logging Database

After creating the database on the database server, the logging database can be setup and configured for use by:

1. Generating the database schema, and applying it the logging database
2. Configuring the configuration logging service to use the new logging database.

The logging database schema can be generated from the Get-LogDBSchema cmdlet, as illustrated below:

```
1 Get-LogDBSchema -DatabaseName "loggingDB"
```



```
                -ServiceGroupName "service group name"
                -ScriptType Database
                -LocalDatabase:$LocalDB
                -DataStore Logging
```

The configuration logging service can be configured to use the logging database with the Set-LogDBConnection cmdlet, as illustrated below:

```
Set-LogDBConnection -DataStore Logging -DBConnection $null
Set-LogDBConnection -DataStore Logging -DBConnection "new logging db
    connection string"
```

Configuration Logging Site Settings

On the Site object:

- The 'State'setting allows configuration logging to be disabled, enabled, or made mandatory.
- The 'Locale'setting specifies the language in which configuration logging data text will be stored.

See Get-LogSite cmdlet help for further information on these settings.

This locale setting applies to the text description that is associated with each log , e.g. 'Create Catalog'. It doesn't apply to other textual information in the log like the names of parameters passed to operations, e.g. 'CatalogName'.

This localisation is applied when the data is logged, and not when the logs are viewed later. For example, logs which are created in English will be displayed in English on an end user system which may be configured with a different locale.

about_Log_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                  # Matches Abc*
Get-<Noun> -Company "Abc`*"                   # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```


Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for

objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings

formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Export-LogReportCsv

March 11, 2024

Exports Configuration Logging data into a CSV file.

Syntax

```
Export-LogReportCsv
      [-OutputFile <String>]
      [-StartDateRange <DateTime>]
      [-EndDateRange <DateTime>]
      [-RawReport]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet exports the Configuration Logging data into a CSV data file. The hierarchical logging data is flattened into a single CSV 'table'. The content of CSV file is not intended to be human-readable. It is meant to be input data for external reporting or manipulation tools (for example, a spread sheet application).

Examples

EXAMPLE 1

Export all logged operations to a csv file.

```
Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv"
```


EXAMPLE 2

Export to a CSV file logged operations started on or after a specified datetime..

```
1  Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv" -
       StartDateRange "2012-12-21 09:00"
```

EXAMPLE 3

Export to a CSV file logged operations started and completed between a date range.

```
1  Export-LogReportCsv -OutputFile "c:\MyReports\LoggingData.csv" -
       StartDateRange "2012-12-21 09:00" -EndDateRange "2012-12-311 18:00"
```

Parameters

-OutputFile

Specifies the path to a file where the CSV data will be saved.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDateRange

Specifies the start time of the earliest operation to include.

Type:	DateTime
Position:	Named
Default value:	DateTime.Min
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-EndDateRange

Specifies the end time of the latest operation to include.

Type:	DateTime
Position:	Named
Default value:	DateTime.UtcNow
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RawReport

Get a CSV raw string report without saving it to a file.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Export-LogReportHtml

Export-LogReportHtml

March 11, 2024

Exports Configuration Logging data into a HTML report.

Syntax

```
Export-LogReportHtml
       [-OutputDirectory <String>]
       [-RawReport]
       [-StartDateRange <DateTime>]
       [-EndDateRange <DateTime>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This cmdlet exports the Configuration Logging data into a HTML report. The report consists of two HTML files:

- Summary.Html - this shows summary information from the high level operation logs.

- Details.Html - this shows additional logging data from the low level operation and operation detail logs.

Hyperlinks in summary.html allow drill-down into the associated low level logging data contained within details.html.

Examples

EXAMPLE 1

Export all logged operations to HTML.

```
Export-LogReportHtml -OutputDirectory "c:\MyReports"
```

EXAMPLE 2

Export to a HTML logged operations started on or after a specified datetime..

```
Export-LogReportHtml -OutputDirectory "c:\MyReports" -StartDateRange "
    2012-12-21 09:00"
```

EXAMPLE 3

Export to HTML logged operations started and completed between a date range.

```
Export-LogReportHtml -OutputDirectory "c:\MyReports" -StartDateRange "
    2012-12-21 09:00" -EndDateRange "2012-12-311 18:00"
```

Parameters

-OutputDirectory

Specifies the path to a directory where the HTML report files will will be saved.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-RawReport

Get a HTML raw string report without saving it to a file.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDateRange

Specifies the start time of the earliest operation to include.

Type:	DateTime
Position:	Named
Default value:	DateTime.Min
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndDateRange

Specifies the end time of the latest operation to include.

Type:	DateTime
Position:	Named
Default value:	DateTime.UtcNow
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Export-LogReportCsv

Get-LogDataStore

March 11, 2024

Gets details for each of the ConfigurationLogging data stores.

Syntax

```
Get-LogDataStore
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns an object for each of the ConfigurationLogging data stores describing the connection string, data store name, db type, provider, schema name, and DB status.

A database connection must be configured in order for this command to be used if the service has a secondary data store. This is not required for the site data store.

Examples

EXAMPLE 1

Get the database connection string for the ConfigurationLogging Service.

```
Get-LogDataStore

ConnectionString : Server=.\SQLEXPRESS;Initial Catalog = databaseName;
    Integrated Security = True
DataStore        : Site
DatabaseType     : SqlServer
Provider         : MSSQL
SchemaName       : LogSiteSchema
Status           : OK

ConnectionString :
DataStore        : Secondary
DatabaseType     : SqlServer
Provider         : MSSQL
SchemaName       : LogSecondarySchema
Status           : DBUnconfigured
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.DataStoreConfiguration

An object describing the connection string, data store name, database type, provider, schema name and database status.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Reset-LogDataStore

Get-LogDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the ConfigurationLogging Service.

Syntax

```
Get-LogDBConnection
    [[-DataStore] <String>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected ConfigurationLogging Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the ConfigurationLogging Service instance running on controller "controller1.mydomain.net".

```
Get-LogDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	2
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current ConfigurationLogging Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the ConfigurationLoggingService has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Set-LogDBConnection
- Get-LogServiceStatus
- Test-LogDBConnection
- Get-LogDataStore

Get-LogDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix ConfigurationLogging Service.

Syntax

```
Get-LogDBSchema
    [-DataStore <String>]
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix ConfigurationLogging Service database schema, add a new ConfigurationLogging service to an existing site, remove a ConfigurationLogging service from a site, or create a database server logon for a ConfigurationLogging service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected ConfigurationLogging service instance, otherwise the scripts relate to ConfigurationLogging service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to ConfigurationLogging service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to ConfigurationLogging service roles

If ScriptType is Evict, the returned script contains:

- Removal of ConfigurationLogging service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix ConfigurationLogging Service and copies it to a file called "C:\ConfigurationLoggingSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a ConfigurationLogging service schema.


```
Get-LogDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\ConfigurationLoggingSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the ConfigurationLogging service. This can be used when configuring a mirror server for use.

```
Get-LogDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    ConfigurationLoggingLogins.sql
```

EXAMPLE 3

Get the full database schema for the secondary data store of the ConfigurationLogging Service and copy it to a file called 'c:\LogSecondarySchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB'that does not already contain a ConfigurationLogging Service secondary schema.

```
Get-LogDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup -
    DataStore Secondary > c:\LogSchema.sql
```

Parameters

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site'or the logical name of the secondary data store.

Type:	String
Position:	Named
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

Specifies the name of the database into which the new ConfigurationLogging service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the ConfigurationLogging services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix ConfigurationLogging Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a ConfigurationLogging Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a ConfigurationLogging Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the ConfigurationLogging Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for ConfigurationLogging services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the ConfigurationLogging Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.
- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogDataStore
- Set-LogDBConnection
- Test-LogDBConnection

Get-LogDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix ConfigurationLogging Service.

Syntax


```
Get-LogDBVersionChangeScript
    [-DataStore <String>]
    -DatabaseName <String>
    -TargetVersion <Version>
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix ConfigurationLogging Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix ConfigurationLogging Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix ConfigurationLogging Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix ConfigurationLogging Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix ConfigurationLogging Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-LogServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-LogDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix ConfigurationLogging Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	Named
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-LogDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix ConfigurationLogging Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shut down during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any ConfigurationLogging services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-LogServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogInstalledDBVersion
- Get-LogServiceStatus
- Get-LogDBSchema

Get-LogHighLevelOperation

March 11, 2024

Gets high level operations

Syntax

```
Get-LogHighLevelOperation
    [-Id <Guid>]
    [-Text <String>]
    [-StartTime <DateTime>]
    [-Source <String>]
    [-EndTime <DateTime>]
    [-IsSuccessful <Boolean>]
    [-User <String>]
    [-AdminMachineIP <String>]
    [-OperationType <OperationType>]
    [-TargetType <String>]
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Retrieves high level operations matching the specified criteria. If no parameters are specified this cmdlet returns all high level operations.

Examples

EXAMPLE 1

Get all high level operations

```
Get-LogHighLevelOperation
```

EXAMPLE 2

Get high level operations which log configuration changes.

```
Get-LogHighLevelOperation -OperationType ConfigurationChange
```

EXAMPLE 3

Get high level operations which log administration activities.

```
Get-LogHighLevelOperation -OperationType AdminActivity
```

EXAMPLE 4

Use advanced filtering to get high level operations with a start time on or after "2013-02-27 09:00:00" and an end time on or before "2013-02-27 18:00:00".

```
Get-LogHighLevelOperation -Filter{
  StartTime -ge "2013-02-27 09:00:00" -and EndTime -le "2013-02-27
     18:00:00" }
```

EXAMPLE 5

Either of these commands will get high level operations which have not yet been completed.

```
Get-LogHighLevelOperation -EndTime $null
Get-LogHighLevelOperation -IsSuccessful $null
```


EXAMPLE 6

Get high level operations performed by user “DOMAIN\UserName”.

```
1  Get-LogHighLevelOperation -User "DOMAIN\UserName"
```

Parameters

-Id

Gets the high level operation with the specified identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Text

Gets high level operations with the specified text

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-StartTime

Gets high level operations with the specified start time

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Source

Gets high level operations with the specified source.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-EndTime

Gets high level operations with the specified end time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsSuccessful

Gets high level operations with the specified success indicator.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-User

Gets high level operations logged by the specified user.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminMachineIP

Gets high level operations logged from the machine with the specified IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OperationType

Gets high level operations with the specified operation type. Values can be:

- AdminActivity - to get operations which log administration activity.
- ConfigurationChange - to get operations which log configuration changes.

Type:	OperationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetType

Gets high level operations with the specified target type.

The target type describes the type of object that was the target of the

configuration change. For example, “Session”or “Machine”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.HighLevelOperation

The returned HighLevelOperation object has the following properties:

- Id - The unique identifier of the operation.
- Text - A description of the operation.
- StartTime - The date and time that the operation started.
- EndTime - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- IsSuccessful - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation never completed.
- User - The identifier of the administrator who performed the operation.
- AdminMachineIP - The IP address of the machine that the operation was initiated from.
- Source - Identifies the XenDesktop console, or custom script, where the operation was initiated from. For example, "Studio", "Desktop Director", "My custom script".
- OperationType - The operation type. This can be 'AdminActivity' or 'ConfigurationChange'.
- TargetTypes - Identifies the type of objects that were affected by the operation. For example, "Catalog" or "Desktop","Machine".
- Parameters - The names and values of the parameters that were supplied for the operation.

Related Links

- Start-LogHighLevelOperation
- Stop-LogHighLevelOperation
- Get-LogLowLevelOperation

Get-LogInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the ConfigurationLogging Service.

Syntax

```
Get-LogInstalledDBVersion
   [-DataStore <String>]
   [-Upgrade]
   [-Downgrade]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the current version number of the Citrix ConfigurationLogging Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-LogDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix ConfigurationLogging Service database schema version number.

```
Get-LogInstalledDBVersion
```

EXAMPLE 2

Get the versions of the ConfigurationLogging Service database schema for which upgrade scripts are supplied.

```
Get-LogInstalledDBVersion -Upgrade
```

Parameters

-DataStore

Specifies the database connection logical name the schema script should be returned for. The parameter is optional.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-LogInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the ConfigurationLogging

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogDBVersionChangeScript
- Get-LogDBSchema

Get-LogLowLevelOperation

March 11, 2024

Gets low level operations

Syntax

```
Get-LogLowLevelOperation
    [-Id <Guid>]
    [-HighLevelOperationId <Guid>]
    [-StartTime <DateTime>]
    [-EndTime <DateTime>]
    [-IsSuccessful <Boolean>]
    [-User <String>]
    [-AdminMachineIP <String>]
    [-Text <String>]
    [-Source <String>]
    [-SourceSdk <String>]
    [-OperationType <OperationType>]
    [-TargetType <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Retrieves low level operations matching the specified criteria. If no parameters are specified this cmdlet returns all low level operations.

Examples

EXAMPLE 1

Get all low level operations

```
Get-LogLowLevelOperation
```

EXAMPLE 2

Get low level operations which log configuration changes.

```
Get-LogLowLevelOperation -OperationType ConfigurationChange
```

EXAMPLE 3

Get low level operations which log administration activities.

```
Get-LogLowLevelOperation -OperationType AdminActivity
```

EXAMPLE 4

Use advanced filtering to get low level operations with a start time on or after "2013-02-27 09:00:00" and an end time on or before "2013-02-27 18:00:00".

```
Get-LogLowLevelOperation -Filter{
  StartTime -ge "2013-02-27 09:00:00" -and EndTime -le "2013-02-27
     18:00:00" }
```

EXAMPLE 5

Either of these commands will get low level operations which have not yet been completed.

```
Get-LogLowLevelOperation -EndTime $null
Get-LogLowLevelOperation -IsSuccessful $null
```


EXAMPLE 6

Get low level operations performed by user “DOMAIN\UserName”.

```
1  Get-LogLowLevelOperation -User "DOMAIN\UserName"
```

Parameters

-Id

Gets the low level operation with the specified identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-HighLevelOperationId

Gets low level operations for the high level operation with the specified identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Gets low level operations with the specified start time

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndTime

Gets low level operations with the specified end time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsSuccessful

Gets low level operations with the specified success indicator.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-User

Gets low level operations logged by the specified administrator.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-AdminMachineIP

Gets low level operations logged from the machine with the specified IP address.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Text

Gets low level operations with the specified text

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Source

Gets low level operations with the specified source.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-SourceSdk

Gets low level operations logged from the SDK with the specified identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OperationType

Gets low level operations with the specified operation type. Values can be:

- AdminActivity - to get operations which log administration activity.
- ConfigurationChange - to get operations which log configuration changes.

Type:	OperationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetType

Gets low level operations with the specified target type. The target type describes the type of object that was the target of the configuration change. For example, “Session”or “Machine”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.LowLevelOperation

The returned LowLevelOperation object has the following properties:

- Id - The unique identifier of the operation.
- Text - A description of the operation.
- HighLevelOperationId - The unique identifier of the related high level operation.
- StartTime - The date and time that the operation started.
- EndTime - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- IsSuccessful - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation never completed.
- AdminSid - The identifier of the administrator who performed the operation.

- AdminMachineIP - The IP address of the machine that the operation was performed on.
- Source - The name of the XenDesktop service that the operation was performed on; for example, “MachineCreation”, “DelegatedAdmin”.
- SourceSdk - The identifier of the XenDesktop service SDK through which the operation was performed; for example, “Prov”, “Admin”.
- OperationType - The operation type. This can be ‘AdminActivity’ or ‘ConfigurationChange’.
- TargetTypes - Identifies the type of objects that were affected by the operation. For example, “Catalog” or “Desktop”,”Machine”.
- Parameters - The names and values of the parameters that were supplied for the operation.
- Details - A collection of OperationDetail objects containing specific information about each object affected by the operation.

Each OperationDetail object in the ‘Details’collection has the following properties:

- TargetUid - The unique identifier of the target object affected by the operation.
- TargetName - The name of the target object affected by the operation.
- TargetType - The type of the target object.
- Text - The description of operation performed on the target object.
- StartTime - The date and time that the operation started.
- EndTime - The date and time that the operation completed. This will be null if the operation is still in progress, or if the operation never completed.
- IsSuccessful - Indicates whether the operation completed successfully or not. This will be null if the operation is still in progress, or if the operation didn’t complete.

The following properties will be set if the operation changed a property on the object:

- PropertyName - The name of the changed property.
- NewValue - The new property value.
- PreviousValue - The previous property value.
- AddValue - If the object property contains a set of values,

  this specifies the new value which was added to the set.
- RemoveValue- If the object property contains a set of values,

  this specifies the value which was removed from the set.

Related Links

- Get-LogHighLevelOperation

Get-LogService

March 11, 2024

Gets the service record entries for the ConfigurationLogging Service.

Syntax

```
Get-LogService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the ConfigurationLogging Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the ConfigurationLogging Service running in the current service group.

```
Get-LogService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Log_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Log_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.Service

The Get-LogServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin

Get-LogServiceAddedCapability

March 11, 2024

Gets any added capabilities for the ConfigurationLogging Service on the controller.

Syntax

```
Get-LogServiceAddedCapability
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables updates to the ConfigurationLogging Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the ConfigurationLogging Service.

```
Get-LogServiceAddedCapability
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin

Get-LogServiceInstance

March 11, 2024

Gets the service instance entries for the ConfigurationLogging Service.

Syntax

```
Get-LogServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the ConfigurationLogging Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the ConfigurationLogging Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
1 Get-LogServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.ServiceInstance

The Get-LogServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Log.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.ConfigurationLogging.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogServiceStatus
- Reset-LogServiceGroupMembership

Get-LogServiceStatus

March 11, 2024

Gets the current state of the ConfigurationLogging Service on the controller.

Syntax

```
Get-LogServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the ConfigurationLogging Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the ConfigurationLogging Service.

```
Get-LogServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-LogServiceStatus command returns an object containing the status of the ConfigurationLogging Service together with extra diagnostics information.

DBUnconfigured

The ConfigurationLogging Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the ConfigurationLogging Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the ConfigurationLogging Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The ConfigurationLogging is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The ConfigurationLogging is connected to a database that is valid, but it does not have the full functionality required so the ConfigurationLogging cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the ConfigurationLogging Service currently in use is incompatible with the version of the ConfigurationLogging Service schema on the database. Upgrade the ConfigurationLogging Service to a more recent version.

DBOlderVersionThanService

The version of the ConfigurationLogging Service schema on the database is incompatible with the version of the ConfigurationLogging Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The ConfigurationLogging Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the ConfigurationLogging Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the ConfigurationLogging and the database has been lost for an extended period of time, or has failed due to a configuration problem. The ConfigurationLogging service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogDataStore
- Set-LogDBConnection
- Test-LogDBConnection
- Get-LogDBConnection
- Get-LogDBSchema

Get-LogSite

March 11, 2024

Gets global configuration logging settings.

Syntax

```
Get-LogSite
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

This cmdlet retrieves the global configuration logging settings.

Examples

EXAMPLE 1

Gets configiration logging site settings.

```
Get-LogSite
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.Site

Get-LogSite returns an object containing the following properties:

- Locale - the current language that logs should be recorded in.

  Can be: 'en', 'ja', 'zh-CN', 'de', 'es'or 'fr'.

- State - the current state of configuration logging. Can be:

  Enabled, Disabled, Mandatory or NotSupported.

Related Links

- Set-LogSite

Get-LogSummary

March 11, 2024

Gets operations logged within time intervals inside a date range.

Syntax

```
Get-LogSummary
   [-StartDateRange <DateTime>]
   [-EndDateRange <DateTime>]
   [-IntervalSeconds <Int64>]
   [-GetLowLevelOperations]
   [-IncludeIncomplete]
   [-OperationType <OperationType>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Get-LogSummary cmdlet retrieves summary counts of operations logged within time intervals inside a date range. The returned data indicates the rate at which configuration changes and activities were performed out within a time period.

Examples

EXAMPLE 1

Get a summary of all completed high level operations. The returned log summary collection will contain a single item for the time period spanning the entire [1900-01-01 00:00:00]-[UtcNow] date range. e.g.

Key ==> Value

01/01/1900 00:00:00 ==> 41

```
$logSummary = Get-LogSummary
```

EXAMPLE 2

Gets a summarised count of completed high level operations logged over two weeks, at daily intervals. The returned log summary collection will contain multiple items; one for each day in the summary date range. - e.g.

Key ==> Value

01/02/2013 14:50:39 ==> 0

02/02/2013 14:50:39 ==> 4

03/02/2013 14:50:39 ==> 21

04/02/2013 14:50:39 ==> 0

05/02/2013 14:50:39 ==> 0

06/02/2013 14:50:39 ==> 0

07/02/2013 14:50:39 ==> 5

08/02/2013 14:50:39 ==> 0

09/02/2013 14:50:39 ==> 0

10/02/2013 14:50:39 ==> 0

11/02/2013 14:50:39 ==> 0

12/02/2013 14:50:39 ==> 7

13/02/2013 14:50:39 ==> 0

14/02/2013 14:50:39 ==> 0

15/02/2013 14:50:39 ==> 12

```
$daily = 60*60*24
[DateTime]$startRange = "2013-02-01 14:50:39"
[DateTime]$endRange = $startRange.AddDays(14)
Get-LogSummary -StartDateRange $startRange -EndDateRange $endRange -
    intervalSeconds $daily
```


EXAMPLE 3

Gets a summarised count of completed low level operations logged during a day, at hourly intervals. The returned log summary collection will contain multiple items; one for each hour in the summary date range - e.g.

Key ==> Value

04/03/2013 00:00:00 ==> 12

04/03/2013 01:00:00 ==> 10

04/03/2013 02:00:00 ==> 5

.

.

.

04/03/2013 21:00:00 ==> 26

04/03/2013 22:00:00 ==> 0

04/03/2013 23:00:00 ==> 9

```
$hourly = 60*60
[DateTime]$startRange = "2013-02-03 00:00:00"
[DateTime]$endRange = "2013-02-03 23:59:59"
Get-LogSummary -StartDateRange $startRange -EndDateRange $endRange -
    intervalSeconds $hourly -GetLowLevelOperations
```

Parameters

-StartDateRange

Specifies the start of the summary period date range

Type:	DateTime
Position:	Named
Default value:	1900-01-01 00:00:00
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndDateRange

Specifies the end of the summary period date range.

Type:	DateTime
Position:	Named
Default value:	DateTime.UtcNow
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IntervalSeconds

Specifies the size, in seconds, of each time interval
required within the summary date range. If this is not specified,
is null, zero or exceeds the specified date range, it defaults to the
total number of seconds between EndDateRange and StartDateRange.

Type:	Int64
Position:	Named
Default value:	Total number of seconds in the EndDateRange and StartDateRange time span.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GetLowLevelOperations

Specifies if the cmdlet should return low level operation summary counts.

Type:	SwitchParameter
Position:	Named

Default value:	$false - high level operations counts are returned.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludeIncomplete

Specifies if incomplete operations should be included

in the returned summary counts.

Type:	SwitchParameter
Position:	Named
Default value:	$false - incomplete operations are excluded.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OperationType

Specifies the type of logged operations to include.

Values can be ‘AdminActivity’or ‘ConfigurationChange’

Type:	OperationType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary<string, int>

The summary data is returned as a collection of dictionary items. The 'Key'value of each dictionary item specifies the start of the time interval within the overall summary date range. The 'Value'data in each dictionary item contains the count of operations which were started within that time interval.

Notes

If the specified summary date range and interval period will result in more than 50,000 intervals being returned Get-LogSummary will generate an error and abort the operation.

Related Links

- Get-LogHighLevelOperation
- Get-LogLowLevelOperation

Remove-LogHighLevelOperationMetadata

March 11, 2024

Removes metadata from the given HighLevelOperation.

Syntax

```
Remove-LogHighLevelOperationMetadata
      -Id <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogHighLevelOperationMetadata
      -Id <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogHighLevelOperationMetadata
      [-InputObject] <HighLevelOperation[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogHighLevelOperationMetadata
      [-InputObject] <HighLevelOperation[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given High Level Operation.

Examples

EXAMPLE 1

Remove all metadata from all HighLevelOperations objects.


```
Get-HighLevelOperation | % {
  Remove-LogHighLevelOperationMetadata -Map $_.MetadataMap  }
```

Parameters

-InputObject

The HighLevelOperations to remove the metadata from

Type:	HighLevelOperation[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Id of the HighLevelOperation.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Set-LogHighLevelOperationMetadata

Remove-LogOperation

March 11, 2024

Deletes configuration logs

Syntax

```
Remove-LogOperation
      -DatabaseCredentials <PSCredential>
      [-StartDateRange <DateTime>]
      [-EndDateRange <DateTime>]
```



```
        [-IncludeIncomplete]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-LogOperation
       -UserName <String>
       -SecurePassword <SecureString>
       [-StartDateRange <DateTime>]
       [-EndDateRange <DateTime>]
       [-IncludeIncomplete]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-LogOperation
       -UserName <String>
       [-Password <String>]
       [-StartDateRange <DateTime>]
       [-EndDateRange <DateTime>]
       [-IncludeIncomplete]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-LogOperation
       -SqlCredentials <PSCredential>
       [-StartDateRange <DateTime>]
       [-EndDateRange <DateTime>]
       [-IncludeIncomplete]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-LogOperation
       -SqlLogin <String>
       -SqlPassword <SecureString>
       [-StartDateRange <DateTime>]
       [-EndDateRange <DateTime>]
       [-IncludeIncomplete]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Remove-LogOperation deletes logs from the Configuration Logging database. All high level operations within the specified time range are removed together with their associated low level operations.

Examples

EXAMPLE 1

Remove all completed operation logs

```
Remove-LogOperation  -UserName "DOMAIN\User"  -Password "UserPassword"
```

EXAMPLE 2

Remove all operations

```
Remove-LogOperation  -UserName "DOMAIN\User"  -Password "UserPassword"
    -IncludeIncomplete
```

EXAMPLE 3

Delete logs started on or after “2023-01-01 12:00:00”and completed on or before “2023-01-31 12:00:00”

```
Remove-LogOperation -username "domain\userName" -password "password" -
    StartDateRange "2023-01-01 12:00:00"  -EndDateRange "2023-01-31
    12:00:00"
```

EXAMPLE 4

Delete logs by supplying Windows user credentials via a credentials object.

```
$securePassword = ConvertTo-SecureString -String "UserPassword" -
    AsPlainText -Force
$credentials    = New-Object -TypeName System.Management.Automation.
    PSCredential -ArgumentList "DOMAIN\UserName", $securePassword
Remove-LogOperation -StartDateRange -DatabaseCredentials $credentials
```

Parameters

-DatabaseCredentials

Specifies the credentials of a Windows user with permission to delete records from the Configuration Logging database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-UserName

Specifies the Windows user name of a database user with permission to delete records from the Configuration Logging database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SecurePassword

Specifies the password a Windows user, in secure string form, with permission to delete records from the Configuration Logging database.

Type:	SecureString
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SqlCredentials

Specifies the SQL credentials of a database user with permission to delete records from the Configuration Logging database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SqlLogin

Specifies the SQL login of a database user with permission to delete records from the Configuration Logging database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SqlPassword

Specifies the SQL password of a database user with permission to delete records from the Configuration Logging database.

Type:	SecureString
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Password

Specifies the password of a Windows user with permission to delete records from the Configuration Logging database.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartDateRange

Specifies the start time of the earliest high level operation to delete

Type:	DateTime
Position:	Named
Default value:	DateTime.Min
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EndDateRange

Specifies the end time of the latest high level operation to delete

Type:	DateTime
Position:	Named

Default value:	DateTime.UtcNow
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludeIncomplete

Specifies if incomplete high level operations should be deleted.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Remove-LogServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-LogServiceMetadata
       [-ServiceHostId] <Guid>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-LogServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-LogService | % {
  Remove-LogServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Set-LogServiceMetadata

Remove-LogSiteMetadata

March 11, 2024

Removes metadata from the given Site.

Syntax

```
Remove-LogSiteMetadata
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogSiteMetadata
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-LogSiteMetadata
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-LogSiteMetadata
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Site.

Examples

EXAMPLE 1

Remove all metadata from all Site objects.

```
Get-LogSite | % {
  Remove-LogSiteMetadata -Map $_.MetadataMap  }
```

Parameters

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Set-LogSiteMetadata

Reset-LogDataStore

March 11, 2024

Refreshes the database string currently being used by the Log service.

Syntax

```
Reset-LogDataStore
      [-DataStore] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Returns the string for the database connection currently being used by the ConfigurationLogging Service. Can only be called for secondary data stores.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Refresh the database connection string for the ConfigurationLogging Service.

```
Reset-LogDataStore -DataStore Secondary
OK
```

Parameters

-DataStore

Specifies the database connection logical name to be used by the ConfigurationLogging Service. Can be either be 'Site'or the logical name of the secondary data store. Specifying the site data store will display an error because this operation is not supported for site data stores.

Type:	String
Position:	2
Default value:	Site
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.ServiceStatus

The status of the specified data store.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogDataStore

Reset-LogEnabledFeatureList

March 11, 2024

Refreshes the ConfigurationLogging service's list of enabled features.

Syntax

```
Reset-LogEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Synchronizes the currently selected Citrix ConfigurationLogging Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected ConfigurationLogging service instance's list of enabled features.

```
1  Reset-LogEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-LogServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the ConfigurationLogging Service.

Syntax

```
Reset-LogServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload ConfigurationLogging Service access permissions and configuration service locations. The Reset-LogServiceGroupMembership command must be run on at least one instance of the service type (Log) after installation and registration with the configuration service. Without this operation, the ConfigurationLogging services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-LogServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    LogServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-LogServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.ConfigurationLogging.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-LogServiceGroupMembership command.

Outputs

Citrix.ConfigurationLogging.Sdk.ServiceInstance

Reset-LogServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the ConfigurationLogging Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogServiceInstance
- Get-LogServiceStatus

Set-LogDBConnection

March 11, 2024

Configures a database connection for the ConfigurationLogging Service.

Syntax

```
Set-LogDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-Force]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix ConfigurationLogging Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-LogDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```


EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
1  Set-LogDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the ConfigurationLogging Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site'or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-LogDBConnection cmdlet returns an object describing the status of the ConfigurationLogging Service together with extra diagnostics information. Possible values are:

- OK:

The ConfigurationLogging Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the ConfigurationLogging Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the ConfigurationLogging Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the ConfigurationLogging Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the ConfigurationLogging Service currently in use is newer than, and incompatible with, the version of the ConfigurationLogging Service schema on the database. Upgrade the ConfigurationLogging Service to a more recent version.

- DBOlderVersionThanService:

The version of the ConfigurationLogging Service schema on the database is newer than, and incompatible with, the version of the ConfigurationLogging Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the ConfigurationLogging Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the ConfigurationLogging Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the ConfigurationLogging Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

DatabaseError

An error occurred in the service while attempting a database operation.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogServiceStatus
- Get-LogDBConnection
- Test-LogDBConnection

Set-LogDBCredentials

March 11, 2024

Configures the database server SQL credentials for the ConfigurationLogging Service.

Syntax

```
Set-LogDBCredentials
    [[-DataStore] <String>]
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```



```
Set-LogDBCredentials
    [[-DataStore] <String>]
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix ConfigurationLogging Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional ConfigurationLogging Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-LogDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-LogDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.


```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-LogDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-LogDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either 'Site' or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-LogDBSchema
- Set-LogDBConnection
- Get-Credential

Set-LogHighLevelOperationMetadata

March 11, 2024

Adds or updates metadata on the given HighLevelOperation.

Syntax

```
Set-LogHighLevelOperationMetadata
   -Id <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogHighLevelOperationMetadata
   -Id <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogHighLevelOperationMetadata
   [-InputObject] <HighLevelOperation[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogHighLevelOperationMetadata
   [-InputObject] <HighLevelOperation[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given HighLevelOperations.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the High Level Operation.

```
Set-LogHighLevelOperationMetadata -Name property -Value value -Id {
 E6DF248A-D2C3-463E-B2BA-7B303D36BB54 }
```


Parameters

-InputObject

The HighLevelOperations to add the metadata to

Type:	HighLevelOperation[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Id of the HighLevelOperation.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the High Level Operation specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Remove-LogHighLevelOperationMetadata

Set-LogServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-LogServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-LogServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1"= "val1"; "name2"= "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

Type:	PSObject
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-LogServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Remove-LogServiceMetadata

Set-LogSite

March 11, 2024

Sets global configuration logging settings.

Syntax

```
Set-LogSite
    [-State <LoggingState>]
    [-Locale <String>]
    [-LoggingDBPurgeDurationDays <Int32>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet sets the global configuration logging settings.

Examples

EXAMPLE 1

Set the logging language to Chinese. The logging state will be unchanged.


```
Set-LogSite -Locale "zh-CN"
```

EXAMPLE 2

Set the logging state to mandatory. The logging locale will be unaffected. In this state, no configuration change will be allowed unless it is successfully logged.

```
Set-LogSite -State "Mandatory"
```

EXAMPLE 3

Set the logging language to German and the state to Enabled.

```
Set-LogSite -Locale "de" -State "Enabled"
```

Parameters

-State

Sets the state of configuration logging. Values can be:

- Enabled - state changes will be logged. If logging is unavailable, the state change will still be permitted.
- Disabled - state changes will not be logged.
- Mandatory - state change will be logged. If logging is unavailable, the state change will not be permitted.
  When the state is set to Enabled or Mandatory, XenDesktop services will attempt to log operations (which perform configuration changes) before performing them.

Type:	LoggingState
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Locale

Sets the language that logs should be recorded in. Values can be: ‘en’, ‘ja’, ‘zh-CN’, ‘de’, ‘es’or ‘fr’ .

Type:	String
Accepted values:	de, en, es, fr, ja, zh-CN
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDBPurgeDurationDays

Sets the number of days for which the configuration logging logs will be retained. Any config logging data older than number of days set in this property will be purged.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.Site

Current logging settings

Notes

Configuration logging will automatically transition to a 'NotSupported'state if the logging feature is not licensed. Set-LogSite will reject request to set logging to 'Enabled'or 'Mandatory'while the logging feature is not licensed.

Related Links

- Get-LogSite

Set-LogSiteMetadata

March 11, 2024

Adds or updates metadata on the given Site.

Syntax

```
Set-LogSiteMetadata
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogSiteMetadata
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-LogSiteMetadata
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

```
Set-LogSiteMetadata
   -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Site objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Site.

```
Set-LogSiteMetadata -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Site specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()’”

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-LogSiteMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Remove-LogSiteMetadata

Start-LogHighLevelOperation

March 11, 2024

Logs the start of a high level operation.

Syntax

```
Start-LogHighLevelOperation
      [-StartTime <DateTime>]
      -Text <String>
      -Source <String>
      [-OperationType <OperationType>]
      [-TargetTypes <String[]>]
      [-Parameters <Hashtable>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Start-LogHighLevelOperation creates a log entry to record the start of a high level operation.

Start-LogHighLevelOperation can be called to log high level configuration changes which originate from customized configuration scripts. Start-LogHighLevelOperation should be called before the script invokes service SDK cmdlets which execute the configuration changes.

Start-LogHighLevelOperation returns a HighLevelOperation object which contains information about the started high level operation. The “Id”property of the returned HighLevelOperation uniquely identifies the started high level operation. This can be supplied into the “-LoggingId”parameter which is implemented in all service SDK cmdlets which execute loggable configuration changes.

High level operation logs are automatically created by the XenDesktop consoles when:

- Initiating an operation which performs configuration changes.
- Initiating an operation which performs an administration activity.

Configuration Change and Administrator Activity

A configuration change is an operation which alters the configuration of the XenDesktop site. Examples of a configuration changes include:

- Creating or editing a host.

- Creating or editing a catalog.
- Adding a user to a delivery group.
- Deleting a machine.

An administrator activity operation doesn't directly alter site configuration, but it could be an operation carried out by an administrator as part of site management or helpdesk support. Examples of administrator activities include:

- Shutdown/start/restart of a user desktop.
- Studio or Director sending a message to a user.
- Rebooting a user's desktop.

Once the change being logged has completed (whether successfully or not) the Stop-LogHighLevelOperation cmdlet should be called to log the completion of a started high level operation.

Examples

EXAMPLE 1

Creates an unmanaged catalog and assigns a machine to it, within the scope of a high level operation start and stop. The identifier of the high level operation is passed into the "-LoggingId"parameter of the service SDK cmdlets. The execution of the cmdlets in the services will create the low level operation logs for the supplied high level operation.

```
$succeeded = $false #indicates if high level operation succeeded.
# Log high level operation start.
$highLevelOp = Start-LogHighLevelOperation -Text "Create catalog" -
    Source "My Custom Script"

try
{

  # Create catalog object
  $catalog = New-BrokerCatalog -Name "MyCatalog" -ProvisioningType
     Manual -AllocationType Permanent -MinimumFunctionalLevel 'LMAX' -
     LoggingId $highLevelOp.Id

  # Add a machine to the catalog
  $machine = New-BrokerMachine -CatalogUid $catalog.Uid -MachineName "
     DOMAIN\Machine" -LoggingId $highLevelOp.Id
  $succeeded = $true
 }

catch{
  "Error encountered"  }


finally{
```



```
21
22    # Log high level operation stop, and indicate its success
23    Stop-LogHighLevelOperation -HighLevelOperationId $highLevelOp.Id -
          IsSuccessful $succeeded
24  }
```

Parameters

-Text

Specifies text to describe the high level operation.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Source

Specifies the source of the high level operation.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTime

Specifies the start time of the high level operation.

Type:	DateTime

Position:	Named
Default value:	DateTime.UtcNow
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OperationType

Specifies the type of operation being logged. Values can be:

- AdminActivity - If the operation being logged performs an administration activity.
- ConfigurationChange - If the operation being logged performs a configuration change.

Type:	OperationType
Position:	Named
Default value:	ConfigurationChange
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetTypes

Specifies the names of the object types that will be affected

by the operation being logged.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Parameters

Specifies the names and values of parameters that are supplied to the operation being logged.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.ConfigurationLogging.Sdk.HighLevelOperation

The newly logged high level operation start.

Related Links

- Stop-LogHighLevelOperation
- Get-LogHighLevelOperation

Stop-LogHighLevelOperation

March 11, 2024

Logs the completion of a previously started high level operation.

Syntax

```
Stop-LogHighLevelOperation
    -HighLevelOperationId <String>
    [-EndTime <DateTime>]
    -IsSuccessful <Boolean>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Stop-LogHighLevelOperation logs the completion of a started high level operation.

Examples

EXAMPLE 1

Creates an unmanaged catalog and assigns a machine to it, within the scope of a high level operation start and stop. The identifier of the high level operation is passed into the "-LoggingId"parameter of the service SDK cmdlets. The execution of the cmdlets in the services will create the low level operation logs for the supplied high level operation.

```
$succeeded = $false #indicates if high level operation succeeded.
# Log high level operation start.
$highLevelOp = Start-LogHighLevelOperation -Text "Create unmanaged
    catalog" -Source "My Custom Script"

try
{

```



```
  # Create catalog object
  $catalog = New-BrokerCatalog -Name "MyCatalog" -ProvisioningType
      Manual -AllocationType Permanent -MinimumFunctionalLevel 'LMAX' -
      LoggingId $highLevelOp.Id

  # Add a machine to the catalog
  $machine = New-BrokerMachine -CatalogUid $catalog.Uid -MachineName "
      DOMAIN\Machine" -LoggingId $highLevelOp.Id
  $succeeded = $true
 }

catch{
  "Error encountered"  }


finally{

  # Log high level operation stop, and indicate its success
  Stop-LogHighLevelOperation -HighLevelOperationId $highLevelOp.Id -
      IsSuccessful $succeeded
 }
```

Parameters

-HighLevelOperationId

Specifies the identifier of the high level operation being completed.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IsSuccessful

Specifies if the started high level operation completed sucessfully.

Type:	Boolean
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-EndTime

Specifies the end time of the high level operation being completed.

Type:	DateTime
Position:	Named
Default value:	DateTime.UtcNow.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Start-LogHighLevelOperation
- Get-LogLowLevelOperation

Test-LogDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix ConfigurationLogging Service.

Syntax

```
Test-LogDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-LogDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-LogDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix ConfigurationLogging Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a ConfigurationLogging SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-LogDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix ConfigurationLogging Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the ConfigurationLogging Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-LogDBConnection cmdlet returns an object describing the status of the selected ConfigurationLogging Service instance that would result if the connection string were used with the Set-LogDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-LogDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the ConfigurationLogging Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the ConfigurationLogging Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The ConfigurationLogging Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The ConfigurationLogging Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the ConfigurationLogging Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the ConfigurationLogging Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseError

An error occurred in the service while attempting a database operation.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_LogConfigurationLoggingSnapin
- Get-LogServiceStatus
- Get-LogDBConnection
- Set-LogDBConnection

about_AdminDelegatedAdminSnapIn

March 11, 2024

Topic

about_AdminDelegatedAdminSnapin

Short Description

The Delegated Administration Service PowerShell snap-in provides administrative functions for the Delegated Administration Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Admin'.

Long Description

The Delegated Administration Service PowerShell snap-in enables both local and remote administration of the Delegated Administration Service.

The Delegated Administration Service (or DAS for short) stores information about Citrix administrators and the rights they have. Services in the XenDesktop deployment use the DAS to determine whether a particular user has the privilege to perform an operation or not.

The snap-in provides storage and configuration of these entities:

Administrators

Each administrator object represents an individual person or a group of people identified by their Active Directory account. Administrators can be enabled and disabled.

The effective rights that a user has is the union of any rights that they have by looking at their Active Directory group membership. Disabled administrator entries are ignored for this calculation.

Once a site is setup, there must always be a full administrator and the Delegated Administration snap-in rejects requests to remove or disable the last full administrator.

Roles

A role represents a job function. That is, anyone with a given role is expected to be able to use or perform the tasks, wizards, and actions associated with that role. Administrators may have multiple roles for a particular site.

Some roles are built-in, and some editions of the product allow custom roles to be created with different combinations of permissions.

Scopes

Scopes represent a collection of objects, and are used to group objects for administrative purposes in a way that is relevant to the organisation. They can be used to represent both hierarchical and non-hierarchical relationships.

Objects can exist in multiple scopes at once. You may find it easier to think of scopes as labels, or a non-exclusive grouping such as a play-list.

All objects are implicitly in the built-in 'All' scope.

Some objects are not scoped, and access to them is through either the 'All' scope or indirectly through a scoped object. For example sessions are not directly scoped but can be accessed using the scope of the desktop group.

The DAS stores information about scopes, but the mapping between scopes and objects is stored and updated using the PowerShell snap-ins of each corresponding service. For example, Delivery Group scopes are managed using the Broker PowerShell snap-in.

Rights

Rights determine what an administrator can do and where they can do it. They are expressed as a number of <role, scope> pairs associated with each administrator.

To gain access to any particular object, a person must match an administrator object that has an appropriate right that allows the required operation in a scope that the object is a member of.

Permissions

Each task, wizard or action in the Citrix Studio or Director consoles represents a unit of functionality that an administrator can perform. Permissions are expressed at a high level and generally correspond directly to the labels in the consoles. For example: "Edit catalog", or "Create delivery group".

Permission groups:

Permissions are grouped into related functionality when displayed by the console.

Operations

Operations are the indivisible unit of functionality that each XenDesktop service can perform, and usually correspond to

individual cmdlets. Internally, each permission requires a number of operations to be performed, possibly by different services.

about_AdminDelegatedAdminSnapIn

March 11, 2024

Topic

about_AdminDelegatedAdminSnapin

Short Description

The Delegated Administration Service PowerShell snap-in provides administrative functions for the Delegated Administration Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Admin'.

Long Description

The Delegated Administration Service PowerShell snap-in enables both local and remote administration of the Delegated Administration Service.

The Delegated Administration Service (or DAS for short) stores information about Citrix administrators and the rights they have. Services in the XenDesktop deployment use the DAS to determine whether a particular user has the privilege to perform an operation or not.

The snap-in provides storage and configuration of these entities:

Administrators

Each administrator object represents an individual person or a group of people identified by their Active Directory account. Administrators can be enabled and disabled.

The effective rights that a user has is the union of any rights that they have by looking at their Active Directory group membership. Disabled administrator entries are ignored for this calculation.

Once a site is setup, there must always be a full administrator and the Delegated Administration snap-in rejects requests to remove or disable the last full administrator.

Roles

A role represents a job function. That is, anyone with a given role is expected to be able to use or perform the tasks, wizards, and actions associated with that role. Administrators may have multiple roles for a particular site.

Some roles are built-in, and some editions of the product allow custom roles to be created with different combinations of permissions.

Scopes

Scopes represent a collection of objects, and are used to group objects for administrative purposes in a way that is relevant to the organisation. They can be used to represent both hierarchical and non-hierarchical relationships.

Objects can exist in multiple scopes at once. You may find it easier to think of scopes as labels, or a non-exclusive grouping such as a play-list.

All objects are implicitly in the built-in 'All' scope.

Some objects are not scoped, and access to them is through either the 'All' scope or indirectly through a scoped object. For example sessions are not directly scoped but can be accessed using the scope of the desktop group.

The DAS stores information about scopes, but the mapping between scopes and objects is stored and updated using the PowerShell snap-ins of each corresponding service. For example, Delivery Group scopes are managed using the Broker PowerShell snap-in.

Rights

Rights determine what an administrator can do and where they can do it. They are expressed as a number of <role, scope> pairs associated with each administrator.

To gain access to any particular object, a person must match an administrator object that has an appropriate right that allows the required operation in a scope that the object is a member of.

Permissions

Each task, wizard or action in the Citrix Studio or Director consoles represents a unit of functionality that an administrator can perform. Permissions are expressed at a high level and generally correspond directly to the labels in the consoles. For example: "Edit catalog", or "Create delivery group".

Permission groups:

Permissions are grouped into related functionality when displayed by the console.

Operations

Operations are the indivisible unit of functionality that each XenDesktop service can perform, and usually correspond to individual cmdlets. Internally, each permission requires a number of operations to be performed, possibly by different services.

about_Admin_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```



```
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces

quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:


```
Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
```


Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Add-AdminPermission

March 11, 2024

Add permissions to the set of permissions of a role.

Syntax

```
Add-AdminPermission
   -Role <String>
   [-InputObject] <Permission[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AdminPermission
   -Role <String>
   [-Permission] <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Add extra permissions to the set of permissions that a role maps to.

Any administrator with a right including that role immediately gains the ability to use the operations of the new permissions.

Duplicate permissions do not produce an error, and permissions are skipped if the role already contains the permission (without error).

You cannot modify the permissions of built-in roles.

You are not permitted to add permissions that you yourself don't have rights to.

Examples

EXAMPLE 1

Add a couple of specific permissions to the 'MyRole'role.

```
Add-AdminPermission -Role MyRole -Permission Global_Read,Logging_Read
```

EXAMPLE 2

Add all of the permissions of the Delivery Administrator role to MyRole.

```
$list = Get-AdminRole "Delivery Administrator" | Select -Expand
    Permissions
Add-AdminPermission -Role MyRole -Permission $list
```


Parameters

-InputObject

Specifies the permissions to add.

Type:	Permission[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Permission

Specifies the list of permissions to add (by identifier).

Type:	String[]
Aliases:	Id
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Role

Role name or identifier of the role to update.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Permission

You can pipe a list of permissions to be added into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Remove-AdminPermission
- Get-AdminPermission
- Get-AdminRole
- Get-AdminPermissionGroup
- Test-AdminAccess

Add-AdminRight

March 11, 2024

Grants a given right to the specified administrator.

Syntax

```
Add-AdminRight
   -Scope <String>
   -Role <String>
   -Administrator <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AdminRight
   -Role <String>
   -Administrator <String>
   [-All]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-AdminRight
   [-InputObject] <Right[]>
   -Administrator <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
6     [<CommonParameters>]
```

Description

Use the Add-AdminRight cmdlet to add rights (role and scope pairs) to an administrator.

For convenience, you can use the -All parameter to specify the 'All'scope.

Use the Get-AdminAdministrator cmdlet to determine what rights an administrator has.

Examples

EXAMPLE 1

Assigns the 'Help Desk Administrator'role and 'London'scope to the 'Admin1'administrator.

```
1 Add-AdminRight -Role 'Help Desk Administrator' -Scope London -
    Administrator DOMAIN\Admin1
```

EXAMPLE 2

Assigns the 'Full Administrator'role and 'All'scope to the 'Admin1'administrator.

```
1 Add-AdminRight -Role 'Full Administrator' -All -Administrator DOMAIN\
    Admin1
```

EXAMPLE 3

Copies the administrator rights from 'ExistingAdmin'to 'NewAdmin'.

```
1 $admin = Get-AdminAdministrator -Name DOMAIN\ExistingAdmin
2 Add-AdminRight -InputObject $admin.Rights -Administrator DOMAIN\
    NewAdmin
```

Parameters

-InputObject

Specifies the rights to add from Right objects.

Type:	Right[]

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Scope

Specifies the scope name or scope identifier.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Role

Specifies the role name or role identifier.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Administrator

Specifies the name or ID of the administrator.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-All

Specifies the 'All'scope. This parameter avoids localization issues or having to type the identifier of the 'All'scope.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Right

You can pipe the rights to be added into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AdminAdministrator
- Get-AdminEffectiveRight
- Remove-AdminRight
- about_AdminDelegatedAdminSnapin

Get-AdminAdministrator

March 11, 2024

Gets administrators configured for this site.

Syntax

```
Get-AdminAdministrator
    [-Sid <String>]
    [[-Name] <String>]
    [-BuiltIn <Boolean>]
    [-Enabled <Boolean>]
    [-IsCloudGroup <Boolean>]
    [-Metadata <String>]
    [-UserIdentity <String>]
    [-UserIdentityType <UserIdentityType>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves administrators matching the specified criteria. If no parameters are specified this cmdlet enumerates all administrators.

See about_Admin_Filtering for information about advanced filtering options.

Examples

EXAMPLE 1

Finds the administrator object (if one exists) for user "DOMAIN\TestUser".

```
Get-AdminAdministrator -Name DOMAIN\TestUser
```

EXAMPLE 2

Finds all disabled administrator objects.

```
Get-AdminAdministrator -Enabled $false
```


Parameters

-Name

Gets administrators with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Sid

Gets administrators with the specified SID (security identifier).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-BuiltIn

Gets administrators that are builtin

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Enabled

Gets administrators with the specified value of Enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsCloudGroup

Gets administrators with the specified cloud group state

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserIdentity

Gets administrators with the specified UserIdentity (Windows security identifier or Citrix Cloud Identity). If the UserIdentityType is a Sid this will be the same as the Sid property. For Cloud administrators this will show cloud identity and the the Sid property will be null.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-UserIdentityType

Gets administrators with the specified UserIdentityType (Windows security identifier or Citrix Cloud Identity).

Type:	UserIdentityType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Administrator

Get-AdminAdministrator returns an object for each matching administrator.

Related Links

- about_Admin_Filtering
- New-AdminAdministrator
- Set-AdminAdministrator
- Remove-AdminAdministrator
- Set-AdminAdministratorMetadata
- Remove-AdminAdministratorMetadata

Get-AdminDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the DelegatedAdmin Service.

Syntax

```
Get-AdminDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected DelegatedAdmin Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the DelegatedAdmin Service instance running on controller "controller1.mydomain.net".

```
Get-AdminDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current DelegatedAdmin Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the DelegatedAdminService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminDBConnection
- Get-AdminServiceStatus
- Test-AdminDBConnection

Get-AdminDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix DelegatedAdmin Service.

Syntax

```
Get-AdminDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Gets SQL scripts that can be used to create a new Citrix DelegatedAdmin Service database schema, add a new DelegatedAdmin service to an existing site, remove a DelegatedAdmin service from a site, or create a database server logon for a DelegatedAdmin service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected DelegatedAdmin service instance, otherwise the scripts relate to DelegatedAdmin service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to DelegatedAdmin service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to DelegatedAdmin service roles

If ScriptType is Evict, the returned script contains:

- Removal of DelegatedAdmin service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix DelegatedAdmin Service and copies it to a file called "C:\DelegatedAdminSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a DelegatedAdmin service schema.

```
Get-AdminDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\DelegatedAdminSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the DelegatedAdmin service. This can be used when configuring a mirror server for use.

```
Get-AdminDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    DelegatedAdminLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new DelegatedAdmin service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the DelegatedAdmin services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix DelegatedAdmin Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a DelegatedAdmin Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a DelegatedAdmin Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the DelegatedAdmin Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for DelegatedAdmin services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the DelegatedAdmin Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminDBConnection
- Test-AdminDBConnection

Get-AdminDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix DelegatedAdmin Service.

Syntax

```
Get-AdminDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix DelegatedAdmin Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix DelegatedAdmin Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix DelegatedAdmin Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix DelegatedAdmin Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix DelegatedAdmin Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-AdminServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-AdminDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix DelegatedAdmin Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-AdminDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix DelegatedAdmin Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shut down during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any DelegatedAdmin services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-AdminServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminInstalledDBVersion
- Get-AdminServiceStatus
- Get-AdminDBSchema

Get-AdminEffectiveAdministrator

March 11, 2024

Retrieve the effective administrator objects for a user.

Syntax

```
Get-AdminEffectiveAdministrator
   [-Name] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This command determines what groups the specified user belongs to and retrieves the matching administrator records. It includes the set of rights that would be granted to the user if he or she used the system.

As this command uses Active Directory to determine what groups the user has, the caller must have the ability to read this information from Active Directory.

Only enabled administrator records are returned.

Examples

EXAMPLE 1

Retrieve the administrator records matching user 'testuser'.


```
Get-AdminEffectiveAdministrator MYDOMAIN\testuser
```

Parameters

-Name

User name or ID of user to query

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

User name or ID of user to query

Outputs

Citrix.DelegatedAdmin.Sdk.Administrator

Administrator records matching the specified user

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminAdministrator

Get-AdminEffectiveRight

March 11, 2024

Gets the set of Right objects associated with the current user.

Syntax

```
Get-AdminEffectiveRight
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Get-AdminEffectiveRight cmdlet returns the effective rights of the current user. This is the union of all rights of the enabled administrators that the current user matches, taking into account Active Directory group membership.

Examples

EXAMPLE 1

Return the effective rights for the current user.

```
Get-AdminEffectiveRight
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Right

The Rights associated with the current user

Related Links

- Get-AdminAdministrator
- Add-AdminRight
- Remove-AdminRight
- about_AdminDelegatedAdminSnapin

Get-AdminInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the DelegatedAdmin Service.

Syntax

```
Get-AdminInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Gets the current version number of the Citrix DelegatedAdmin Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-AdminDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix DelegatedAdmin Service database schema version number.

```
1  Get-AdminInstalledDBVersion
```

EXAMPLE 2

Get the versions of the DelegatedAdmin Service database schema for which upgrade scripts are supplied.

```
1  Get-AdminInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-AdminInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the DelegatedAdmin

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminDBVersionChangeScript
- Get-AdminDBSchema

Get-AdminPermission

March 11, 2024

Gets permissions configured for the site.

Syntax

```
Get-AdminPermission
    [-Id <String>]
    [[-Name] <String>]
    [-Description <String>]
    [-GroupId <String>]
    [-GroupName <String>]
    [-IsHidden <Boolean>]
    [-Metadata <String>]
    [-Operation <String>]
    [-ReadOnly <Boolean>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves permission matching the specified criteria. If no parameters are specified this cmdlet enumerates all permissions.

Permissions are configured using the Import-AdminRoleConfiguration command, and are used to represent collections of operations that are needed to perform a particular task. These permissions are presented in Citrix Studio when configuring roles.

Permissions can also have metadata associated with them.

See about_Admin_Filtering for information about advanced filtering options.

Examples

EXAMPLE 1

Finds all permissions with 'Edit'in their names.

```
Get-AdminPermission -Name *Edit*
```

EXAMPLE 2

Finds permissions that contain specific operations, with the -Filter form needed to match multiple values.

```
Get-AdminPermission -Operation "Broker:SetCatalog"
Get-AdminPermission -Filter {
  Operations -contains "Broker:SetCatalog" -or Operations -contains "
     Broker:NewCatalog"  }
```

Parameters

-Name

Gets permissions with the specified name (localized)

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

Gets permissions with the specified identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Gets permissions with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-GroupId

Gets permissions that are a member of the specified permission group (by group id).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-GroupName

Gets permissions that are a member of the specified permission group (by group name).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsHidden

Gets permissions with the specified value for the IsHidden flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Operation

Gets permissions that contain a specific operation.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReadOnly

Gets permissions with the specified value for the ReadOnly flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Permission

Get-AdminPermission returns an object for each matching permission.

Related Links

- Add-AdminPermission
- Remove-AdminPermission
- Get-AdminRole
- Get-AdminPermissionGroup
- Test-AdminAccess

Get-AdminPermissionGroup

March 11, 2024

Gets permission groups configured for the site.

Syntax

```
Get-AdminPermissionGroup
    [-Id <String>]
    [[-Name] <String>]
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves permission groups matching the specified criteria. If no parameters are specified this cmdlet enumerates all permission groups.

Permission groups are configured using the Import-AdminRoleConfiguration command, and are primarily used to store the localized name for a group of permissions. Permission groups can also have metadata associated with them.

See about_Admin_Filtering for information about advanced filtering options.

Examples

EXAMPLE 1

Finds all permission groups that end with the letter 's'.

```
Get-AdminPermissionGroup -Name *s
```

EXAMPLE 2

Retrieve two specific permission group objects with the matching identifiers.

```
$list = @("Hosts","Other")
Get-AdminPermissionGroup -Filter {
  Id -in $list  }
  -SortBy Name
```

Parameters

-Name

Gets permission groups matching the given name. This is the localized name of the group.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

Gets permission groups with the given identifier. This is the non-localized identifier used to associate permissions with permission groups.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.PermissionGroup

Get-AdminPermissionGroup returns an object for each matching permission group.

Related Links

- about_Admin_Filtering
- Import-AdminRoleConfiguration
- Get-AdminPermission

Get-AdminRevision

March 11, 2024

Gets the current revision of the delegated administration configuration data.

Syntax

```
Get-AdminRevision
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Delegated Administration Service maintains a revision number that is incremented whenever its configuration is changed. This command retrieves the current revision number.

Examples

EXAMPLE 1

Retrieve the current revision number.

```
Get-AdminRevision
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Int32

The configuration revision number

Notes

If Int32.MaxValue is reached the value of the revision number will wrap around to Int32.MinValue.

In some cases the value may be incremented by more than one.

Related Links

- about_AdminDelegatedAdminSnapin

Get-AdminRole

March 11, 2024

Gets roles configured for this site.

Syntax

```
Get-AdminRole
    [-Id <Guid>]
    [[-Name] <String>]
    [-BuiltIn <Boolean>]
    [-CanLaunchManage <Boolean>]
    [-CanLaunchMonitor <Boolean>]
    [-Description <String>]
    [-ExclusivePermission <String>]
    [-Metadata <String>]
    [-Permission <String>]
    [-Property <String[]>]
```



```
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves roles matching the specified criteria. If no parameters are specified, this cmdlet enumerates all roles.

See about_Admin_Filtering for information about advanced filtering options.

Examples

EXAMPLE 1

Gets the details of the role with the specific id.

```
Get-AdminRole -Id 20852cdf-f527-4953-ba6e-e7545217122d
```

EXAMPLE 2

List all custom roles.

```
Get-AdminRole -BuiltIn $false
```

Parameters

-Name

Gets roles with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

Gets the role with the specified identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-BuiltIn

Gets roles with the specified value of the BuiltIn flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CanLaunchManage

Gets roles with the specified value of the CanLaunchManage flag.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CanLaunchMonitor

Gets roles with the specified value of the CanLaunchMonitor flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Gets roles with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ExclusivePermission

Gets roles that contain a specific exclusive permission.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Permission

Gets roles that contain a specific permission.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Role

Get-AdminRole returns an object for each matching role.

Related Links

- about_Admin_Filtering
- New-AdminRole
- Set-AdminRole
- Rename-AdminRole
- Remove-AdminRole
- Set-AdminRoleMetadata
- Remove-AdminRoleMetadata

Get-AdminRoleConfiguration

March 11, 2024

Gets role configurations for this site.

Syntax

```
Get-AdminRoleConfiguration
    [-Id <Guid>]
    [[-Name] <String>]
    [-Locale <String>]
    [-Priority <Int32>]
    [-Version <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Retrieves role configurations matching the specified criteria. If no parameters are specified, this cmdlet enumerates and returns all role configurations.

Role configurations are part of the product configuration and define what permissions, permission groups, and built-in roles the product has. This cmdlet also provides the mapping of permissions to operations.

Examples

EXAMPLE 1

Retrieve the role configuration for the Citrix Director component.

```
1  Get-AdminRoleConfiguration -Name Director
```

Parameters

-Name

Gets role configurations matching the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

Gets role configurations with the specified id.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Locale

Gets role configurations with the specified locale. Role configurations usually have a consistent locale.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Priority

Gets role configurations with the specified priority.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Version

Gets role configurations with the matching version number.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.RoleConfiguration

Get-AdminRoleConfiguration returns an object for each matching role configuration.

Notes

This command is supplied for infrastructure purposes only and is not intended for public use.

Related Links

- about_Admin_Filtering
- Import-AdminRoleConfiguration

Get-AdminScope

March 11, 2024

Gets scopes configured for this site.

Syntax


```
Get-AdminScope
    [-Id <Guid>]
    [[-Name] <String>]
    [-BuiltIn <Boolean>]
    [-Description <String>]
    [-Metadata <String>]
    [-TenantId <String>]
    [-TenantName <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves scopes matching the specified criteria. If no parameters are specified this cmdlet enumerates all scopes.

There is one special built-in scope, the 'All'scope.

To determine what objects are currently in a scope, use the Get-<Prefix>ScopedObject from each of the relevant PowerShell snap-ins.

See about_Admin_Filtering for information about advanced filtering options.

Examples

EXAMPLE 1

List all scopes that contain the word 'Sales'.

```
Get-AdminScope -Name *Sales*
```

EXAMPLE 2

Gets the details of the scope with the specific id.

```
Get-AdminScope -Id 21862daf-e529-4553-ba6e-f7543217111e
```


Parameters

-Name

Gets scopes with the specified name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

Gets the scope with the specified identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-BuiltIn

Gets scopes with the specified value of the BuiltIn flag.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Description

Gets scopes with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantId

Gets scopes with the specified tenant customer identifier.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-TenantName

Gets scopes with the specified tenant customer name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline.

See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Scope

Get-AdminScope returns an object for each matching scope.

Related Links

- about_Admin_Filtering
- New-AdminScope
- Set-AdminScope
- Rename-AdminScope
- Remove-AdminScope
- Set-AdminScopeMetadata
- Remove-AdminScopeMetadata

Get-AdminService

March 11, 2024

Gets the service record entries for the DelegatedAdmin Service.

Syntax

```
Get-AdminService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the DelegatedAdmin Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the DelegatedAdmin Service running in the current service group.

```
Get-AdminService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
```



```
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Service

The Get-AdminServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin

Get-AdminServiceAddedCapability

March 11, 2024

Gets any added capabilities for the DelegatedAdmin Service on the controller.

Syntax

```
Get-AdminServiceAddedCapability
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables updates to the DelegatedAdmin Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the DelegatedAdmin Service.

```
Get-AdminServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin

Get-AdminServiceInstance

March 11, 2024

Gets the service instance entries for the DelegatedAdmin Service.

Syntax

```
Get-AdminServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the DelegatedAdmin Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the DelegatedAdmin Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-AdminServiceInstance
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.ServiceInstance

The Get-AdminServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Admin.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.DelegatedAdmin.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminServiceStatus
- Reset-AdminServiceGroupMembership

Get-AdminServiceStatus

March 11, 2024

Gets the current state of the DelegatedAdmin Service on the controller.

Syntax

```
Get-AdminServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Enables the status of the DelegatedAdmin Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the DelegatedAdmin Service.

```
1  Get-AdminServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-AdminServiceStatus command returns an object containing the status of the DelegatedAdmin Service together with extra diagnostics information.

DBUnconfigured

The DelegatedAdmin Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the DelegatedAdmin Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the DelegatedAdmin Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The DelegatedAdmin is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The DelegatedAdmin is connected to a database that is valid, but it does not have the full functionality required so the DelegatedAdmin cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the DelegatedAdmin Service currently in use is incompatible with the version of the DelegatedAdmin Service schema on the database. Upgrade the DelegatedAdmin Service to a more recent version.

DBOlderVersionThanService

The version of the DelegatedAdmin Service schema on the database is incompatible with the version of the DelegatedAdmin Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The DelegatedAdmin Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the DelegatedAdmin Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the DelegatedAdmin and the database has been lost for an extended period of time, or has failed due to a configuration problem. The DelegatedAdmin service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminDBConnection
- Test-AdminDBConnection
- Get-AdminDBConnection
- Get-AdminDBSchema

Get-AdminSite

March 11, 2024

Gets the site level data for the admin service

Syntax

```
Get-AdminSite
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves the site level data such as when the most recent admin sync happened and the current database revision

Examples

EXAMPLE 1

Retrieves the site level data for the admin service such as when the recent admin sync happened and the current database revision

```
Get-AdminSite
```

Parameters

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Admin_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Admin_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Site

Get-AdminSite returns the site level data such as when the most

recent admin sync happened and the current database revision

Related Links

- Get-AdminSite

Import-AdminRoleConfiguration

March 11, 2024

Imports role configuration data into the Delegated Administration Service.

Syntax

```
Import-AdminRoleConfiguration
      [-Path] <String>
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Import-AdminRoleConfiguration
       -Content <String>
       [-Force]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This command is intended for use by Citrix Studio to import definitions, roles, permissions, and their mappings to operations.

The supplied configuration requires a digital signature; this is used to validate the integrity of the configuration.

Examples

EXAMPLE 1

Imports the contents of 'C:\MyAdminConfig.xml'to the Delegated Administration Service.

```
Import-AdminRoleConfiguration -Path 'C:\MyAdminConfig.xml'
```

Parameters

-Path

The path to the file containing the role configuration data.

Type:	String
Aliases:	PSPath
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Content

The content of the role configuration data.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Force

Allows older versions of role configuration to replace newer versions.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

This command is supplied for infrastructure purposes only and is not intended for public use.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminRoleConfiguration

New-AdminAdministrator

March 11, 2024

Adds a new administrator to the site.

Syntax

```
New-AdminAdministrator
   [-Enabled <Boolean>]
   [-IsHidden <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AdminAdministrator
   [-Enabled <Boolean>]
   [-IsHidden <Boolean>]
   [-Name] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-AdminAdministrator
   [-Enabled <Boolean>]
   [-IsHidden <Boolean>]
   -Sid <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

New-AdminAdministrator creates a new administrator object in the site. Once a new administrator has been created you can assign rights (role and scope pairs) to the administrator.

Administrator objects are used to determine what rights, and therefore what permissions a particular Active Directory user has through the various SDKs and consoles of the site.

When the Enabled flag of an administrator is set to false, any rights of the administrator are ignored by the system when performing permission checks.

Examples

EXAMPLE 1

Creates a new administrator object for user "DOMAIN\TestUser". It defaults to enabled.

```
New-AdminAdministrator -Name DOMAIN\TestUser
```

EXAMPLE 2

Creates a new administrator object for user with SID "S-1-2-34-1234567890-1234567890-1234567890-123". It defaults to enabled.

```
New-AdminAdministrator -Sid S
    -1-2-34-1234567890-1234567890-1234567890-123
```

Parameters

-Name

Specifies the user or group name in Active Directory that this administrator corresponds to.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Sid

Specifies the SID (security identifier) of the user in Active Directory that this administrator corresponds to.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Enabled

Specifies whether the new administrator starts off enabled or not.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IsHidden

Specifies whether the administrator is hidden or not.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Administrator

The newly created administrator.

Related Links

- Get-AdminAdministrator
- Set-AdminAdministrator
- Remove-AdminAdministrator

- Set-AdminAdministratorMetadata
- Remove-AdminAdministratorMetadata

New-AdminRole

March 11, 2024

Adds a new custom role to the site.

Syntax

```
New-AdminRole
   [-CanLaunchManage <Boolean>]
   [-CanLaunchMonitor <Boolean>]
   [-Description <String>]
   [-Name] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

New-AdminRole adds a new custom role object to the site. Once a new role has been created, you can add permissions to the role which define what operations the role conveys.

Roles represent a job function, such as 'help desk administrator', and contain a list of permissions that are required to perform that job function.

To assign a role to an administrator, you combine it with a scope which indicates what objects the role can operate on. This pair (also known as a 'right') can then be assigned to an administrator. See Add-AdminRight for further details.

The identifier of the new role is chosen automatically, and custom roles created with this cmdlet always have their BuiltIn flag set to false.

You cannot modify built-in roles.

The following license editions are supported for custom roles:

- Citrix DaaS supports custom roles in all editions.
- Citrix Virtual Apps and Desktops supports custom roles in Premium and Advanced editions.
- Citrix Virtual Apps supports custom roles in Premium and Advanced editions.

Examples

EXAMPLE 1

Creates a new role called 'Supervisor', and then copies the permissions from the help desk role and adds some extras. Then gives this role (with the all scope) to user 'TestUser'.

```
New-AdminRole -Name Supervisor -Description "My custom supervisor role"
$list = Get-AdminRole 'Help Desk Administrator' | Select -Expand
    Permissions
Add-AdminPermission -Role Supervisor -Permission $list
Add-AdminPermission -Role Supervisor -Permission $extras
Add-AdminRight -Administrator DOMAIN\TestUser -Role Supervisor -All
```

Parameters

-Name

Specifies the name of the role. Each role in a site must have a unique name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CanLaunchManage

Optionally Specifies whether the role has access to the Manage tab in Citrix Cloud.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-CanLaunchMonitor

Optionally Specifies whether the role has access to the Monitor tab in Citrix Cloud.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Specifies the description of the role.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Role

The newly created role.

Notes

Roles are created without any permissions. Use the Add-AdminPermission to add permissions.

Related Links

- Get-AdminRole
- Set-AdminRole
- Rename-AdminRole
- Remove-AdminRole

- Set-AdminRoleMetadata
- Remove-AdminRoleMetadata
- Add-AdminPermission
- Add-AdminRight

New-AdminScope

March 11, 2024

Adds a new scope to the site.

Syntax

```
New-AdminScope
    [-Description <String>]
    [-Name] <String>
    [-TenantId <String>]
    [-TenantName <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

New-AdminScope adds a new scope object to the site.

A scope represents a collection of objects. Scopes are used to group objects in a way that is relevant to the organization; for example, the set of delivery groups used by the Sales team.

You can create objects in particular scopes by specifying the -Scope parameter of a New- cmdlet for an object that can be scoped. You can then modify the contents of a scope with Add-<Noun>Scope and Remove-<Noun>Scope cmdlets from the correpsonding PowerShell snap-ins.

To assign a scope to an administrator, combine it with a role and then assign this pair (also known as a 'right') to an administrator. See Add-AdminRight for further details.

The identifier of the new scope is chosen automatically.

Examples

EXAMPLE 1

Creates a new scope called 'Sales', adds a hypervisor connection object to the scope, and then assigns the right to use the hosting role on the Sales scope to the 'TestUser'administrator.

```
New-AdminScope -Name Sales -Description "Sales department scope"
Add-HypHypervisorConnectionScope -HypervisorConnectionName XenServer2 -
    Scope Sales
Add-AdminRight -Administrator DOMAIN\TestUser -Role Hosting -Scope
    Sales
```

Parameters

-Name

Specifies the name of the scope. Each scope in a site must have a unique name.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Specifies the description of the scope.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-TenantId

Specifies the tenant customer identifier of the scope.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-TenantName

Specifies the new tenant name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.Scope

The newly created scope.

Related Links

- Get-AdminScope
- Set-AdminScope
- Rename-AdminScope
- Remove-AdminScope
- Set-AdminScopeMetadata
- Remove-AdminScopeMetadata
- Add-AdminRight

Remove-AdminAdministrator

March 11, 2024

Removes administrators from the site.

Syntax

```
Remove-AdminAdministrator
      [-InputObject] <Administrator[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminAdministrator
      -Sid <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminAdministrator
      [-Name] <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-AdminAdministrator cmdlet deletes administrators from the site.

Examples

EXAMPLE 1

Remove the administrator called “DOMAIN\TestUser”.

```
Remove-AdminAdministrator DOMAIN\TestUser
```

EXAMPLE 2

Remove all disabled administrators.

```
Get-AdminAdministrator -Enabled $false | Remove-AdminAdministrator
```


Parameters

-InputObject

Specifies the administrators to delete.

Type:	Administrator[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the name of the administrator to delete.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Sid

Specifies the SID of the administrator to delete.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Administrator

You can pipe the adminstrators to be deleted into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-AdminAdministrator
- Get-AdminAdministrator
- Set-AdminAdministrator
- Set-AdminAdministratorMetadata
- Remove-AdminAdministratorMetadata

Remove-AdminAdministratorMetadata

March 11, 2024

Removes metadata from the given Administrator.

Syntax

```
Remove-AdminAdministratorMetadata
       -AdministratorSid <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminAdministratorMetadata
       -AdministratorSid <String>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminAdministratorMetadata
       [-AdministratorName] <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-AdminAdministratorMetadata
      [-AdministratorName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminAdministratorMetadata
      [-InputObject] <Administrator[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminAdministratorMetadata
      [-InputObject] <Administrator[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Administrator.

Examples

EXAMPLE 1

Remove all metadata from all Administrator objects.

```
Get-AdminAdministrator | % {
  Remove-AdminAdministratorMetadata -Map $_.MetadataMap  }
```

Parameters

-AdministratorName

Name of the Administrator

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Administrator[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-AdministratorSid

Sid of the Administrator

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Name

The metadata property to remove.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminAdministratorMetadata

Remove-AdminPermission

March 11, 2024

Remove permissions from the set of permissions of a role.

Syntax

```
Remove-AdminPermission
      -Role <String>
      [-InputObject] <Permission[]>
      [-LoggingId <Guid>]
```



```
5         [<CitrixCommonParameters>]
6         [<CommonParameters>]
```

```
1 Remove-AdminPermission
2       -Role <String>
3       [-Permission] <String[]>
4       [-LoggingId <Guid>]
5       [<CitrixCommonParameters>]
6       [<CommonParameters>]
```

```
1 Remove-AdminPermission
2       -Role <String>
3       [-All]
4       [-LoggingId <Guid>]
5       [<CitrixCommonParameters>]
6       [<CommonParameters>]
```

Description

Remove permissions from the set of permissions that a role maps to.

Any administrator with a right including that role immediately loses the ability to use the operations of the removed permissions.

Duplicate permissions do not produce an error, and permissions that the roles does not already have are skipped (without error).

You cannot modify the permissions of built-in roles.

Examples

EXAMPLE 1

Remove a couple of specific permissions from the 'MyRole'role.

```
1 Remove-AdminPermission -Role MyRole -Permission Global_Read,
     Logging_Read
```

EXAMPLE 2

Remove all permissions from the 'MyRole'role.

```
1 Remove-AdminPermission -Role MyRole -All
```


Parameters

-InputObject

Specifies the permissions to remove.

Type:	Permission[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Permission

Specifies the list of permissions to remove (by identifier).

Type:	String[]
Aliases:	Id
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Role

Role name or identifier of the role to update.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-All

Remove all permissions.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Permission

You can pipe a list of permissions to be removed into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Add-AdminPermission
- Get-AdminPermission
- Get-AdminRole
- Get-AdminPermissionGroup
- Test-AdminAccess

Remove-AdminRight

March 11, 2024

Removes rights from an administrator.

Syntax

```
Remove-AdminRight
      -Scope <String>
      -Role <String>
      -Administrator <String>
```



```
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-AdminRight
       -Role <String>
       -Administrator <String>
       [-All]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRight
       [-InputObject] <Right[]>
       -Administrator <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This command removes rights from the specified administrator.

For convenience, you can use the -All parameter to specify the 'All'scope.

Use the Get-AdminAdministrator cmdlet to determine what rights an administrator has.

Examples

EXAMPLE 1

Removes the 'Help Desk Administrator'role and 'London'scope from user 'Admin1'

```
RemoveAdminRight -Role 'Help Desk Administrator' -Scope London -
    Administrator DOMAIN\Admin1
```

EXAMPLE 2

Removes all rights from administrator 'Admin'.

```
$admin = Get-AdminAdministrator -Name DOMAIN\Admin
Remove-AdminRight -InputObject $admin.Rights -Administrator DOMAIN\
    Admin
```


Parameters

-InputObject

Specifies the rights to remove.

Type:	Right[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Scope

Specifies the scope name or scope identifier.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Role

Specifies the role name or role identifier.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Administrator

Specifies the name or ID of the administrator.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-All

Specifies the 'All'scope. This parameter avoids localization issues or having to type the identifier of the 'All'scope.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Right

You can pipe the rights to be removed into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AdminAdministrator
- Get-AdminEffectiveRight
- Add-AdminRight
- about_AdminDelegatedAdminSnapin

Remove-AdminRole

March 11, 2024

Removes a role from the site.

Syntax

```
Remove-AdminRole
       [-InputObject] <Role[]>
       [-Force]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRole
       [-Id] <Guid[]>
       [-Force]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRole
       [-Name] <String[]>
       [-Force]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Remove-AdminRole cmdlet deletes roles from the site.

You cannot remove built-in roles.

An error will be produced if the role being removed is currently assigned to an administrator unless you specify the -Force option. When -Force is specified, any rights that reference the role are also removed.

Examples

EXAMPLE 1

Remove the Supervisor role.


```
Remove-AdminRole -Name Supervisor
```

EXAMPLE 2

Attempt to remove all custom roles. This fails if one of the roles is assigned to an administrator.

```
Get-AdminRole -BuiltIn $false | Remove-AdminRole
```

Parameters

-InputObject

Specifies the roles to remove (by role object).

Type:	Role[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the roles to remove (by role id).

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the roles to remove (by role name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Force

Allow removal of roles that are still in use.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Role

You can pipe the roles to be deleted into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-AdminRole
- Get-AdminRole
- Set-AdminRole
- Rename-AdminRole
- Set-AdminRoleMetadata
- Remove-AdminRoleMetadata

Remove-AdminRoleMetadata

March 11, 2024

Removes metadata from the given Role.

Syntax

```
Remove-AdminRoleMetadata
       [-RoleId] <Guid>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRoleMetadata
       [-RoleId] <Guid>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRoleMetadata
       [-RoleName] <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRoleMetadata
       [-RoleName] <String>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRoleMetadata
       [-InputObject] <Role[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminRoleMetadata
       [-InputObject] <Role[]>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
```



```
6         [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Role.

Examples

EXAMPLE 1

Remove all metadata from all Role objects.

```
1 Get-AdminRole | % {
2   Remove-AdminRoleMetadata -Map $_.MetadataMap  }
```

Parameters

-RoleId

Id of the Role

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-RoleName

Name of the Role

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Role[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminRoleMetadata

Remove-AdminScope

March 11, 2024

Removes a scope from the site.

Syntax

```
Remove-AdminScope
      [-InputObject] <Scope[]>
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminScope
      [-Id] <Guid[]>
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminScope
      [-Name] <String[]>
      [-Force]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

The Remove-AdminScope cmdlet deletes scopes from the site.

You cannot remove the built-in 'All'scope.

An error will be produced if the scope being removed is currently assigned to an administrator unless you specify the -Force option. When -Force is specified, any rights that reference the scope are also removed.

Examples

EXAMPLE 1

Remove the Sales scope.

```
1  Remove-AdminScope -Name Sales
```

EXAMPLE 2

Attempt to remove all scopes (excluding the built-in 'All'scope). This fails if one of the scopes is assigned to an administrator.

```
1  Get-AdminScope -BuiltIn $false | Remove-AdminScope
```

Parameters

-InputObject

Specifies the scopes to remove (by scope object).

Type:	Scope[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the scopes to remove (by scope id).

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the scopes to remove (by scope name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Force

Allow removal of scopes that are still in use.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Scope

You can pipe the scopes to be deleted into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-AdminScope
- Get-AdminScope
- Set-AdminScope
- Rename-AdminScope
- Set-AdminScopeMetadata
- Remove-AdminScopeMetadata

Remove-AdminScopeMetadata

March 11, 2024

Removes metadata from the given Scope.

Syntax

```
Remove-AdminScopeMetadata
       [-ScopeId] <Guid>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminScopeMetadata
       [-ScopeId] <Guid>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminScopeMetadata
       [-ScopeName] <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminScopeMetadata
       [-ScopeName] <String>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Remove-AdminScopeMetadata
      [-InputObject] <Scope[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminScopeMetadata
      [-InputObject] <Scope[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Scope.

Examples

EXAMPLE 1

Remove all metadata from all Scope objects.

```
Get-AdminScope | % {
  Remove-AdminScopeMetadata -Map $_.MetadataMap  }
```

Parameters

-ScopeId

Id of the Scope

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ScopeName

Name of the Scope

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Scope[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminScopeMetadata

Remove-AdminServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-AdminServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-AdminServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-AdminServiceMetadata
       [-InputObject] <Service[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-AdminServiceMetadata
       [-InputObject] <Service[]>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-AdminService | % {
  Remove-AdminServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Set-AdminServiceMetadata

Rename-AdminRole

March 11, 2024

Rename a role

Syntax

```
Rename-AdminRole
       [-InputObject] <Role>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-AdminRole
       [-Id] <Guid>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-AdminRole
       [-Name] <String>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

The Rename-AdminRole cmdlet changes the name of a role.

Role names must be unique, and you cannot modify the name of built-in roles.

Examples

EXAMPLE 1

Renames the 'Supervisor'role to 'HelpDeskLead'.

```
Rename-AdminRole -Name Supervisor -NewName HelpDeskLead
```

Parameters

-InputObject

Specifies the role to rename (by object).

Type:	Role
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the role to rename (by id).

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the role to rename (by name).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the role.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Role

You can pipe the role to be renamed into this command.

Outputs

None or Citrix.DelegatedAdmin.Sdk.Role

When you use the PassThru parameter, Rename-AdminRole generates a Citrix.DelegatedAdmin.Sdk.Role object. Otherwise, this cmdlet does not generate any output.

Related Links

- New-AdminRole
- Get-AdminRole
- Set-AdminRole
- Remove-AdminRole
- Set-AdminRoleMetadata
- Remove-AdminRoleMetadata

Rename-AdminScope

March 11, 2024

Rename a scope

Syntax

```
Rename-AdminScope
       [-InputObject] <Scope>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-AdminScope
       [-Id] <Guid>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Rename-AdminScope
       [-Name] <String>
       [-NewName] <String>
       [-PassThru]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

The Rename-AdminScope cmdlet changes the name of a scope.

Scope names must be unique, and you cannot modify the name of the built-in 'All'scope.

Examples

EXAMPLE 1

Renames the 'Sales'scope to 'SalesDesktops'.

```
Rename-AdminScope -Name Sales -NewName SalesDesktops
```

Parameters

-InputObject

Specifies the scope to rename (by object).

Type:	Scope
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the scope to rename (by id).

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-Name

Specifies the scope to rename (by name).

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NewName

Specifies the new name of the scope.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Scope

You can pipe the scope to be renamed into this command.

Outputs

None or Citrix.DelegatedAdmin.Sdk.Scope

When you use the PassThru parameter, Rename-AdminScope generates a Citrix.DelegatedAdmin.Sdk.Scope object. Otherwise, this cmdlet does not generate any output.

Related Links

- New-AdminScope
- Get-AdminScope
- Set-AdminScope
- Remove-AdminScope
- Set-AdminScopeMetadata
- Remove-AdminScopeMetadata

Reset-AdminEnabledFeatureList

March 11, 2024

Refreshes the DelegatedAdmin service's list of enabled features.

Syntax

```
Reset-AdminEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix DelegatedAdmin Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected DelegatedAdmin service instance's list of enabled features.

```
Reset-AdminEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-AdminServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the DelegatedAdmin Service.

Syntax

```
Reset-AdminServiceGroupMembership
     [-ConfigServiceInstance] <ServiceInstance[]>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Enables you to reload DelegatedAdmin Service access permissions and configuration service locations. The Reset-AdminServiceGroupMembership command must be run on at least one instance of the service type (Admin) after installation and registration with the configuration service. Without this operation, the DelegatedAdmin services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-AdminServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    AdminServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-AdminServiceGroupmembership
```


Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-AdminServiceGroupMembership command.

Outputs

Citrix.DelegatedAdmin.Sdk.ServiceInstance

Reset-AdminServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the DelegatedAdmin Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminServiceInstance
- Get-AdminServiceStatus

Set-AdminAdministrator

March 11, 2024

Sets the properties of an administrator.

Syntax

```
Set-AdminAdministrator
   [-InputObject] <Administrator[]>
   [-Enabled <Boolean>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministrator
   -Sid <String[]>
   [-Enabled <Boolean>]
```



```
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminAdministrator
    [-Name] <String[]>
    [-Enabled <Boolean>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-AdminAdministrator cmdlet is used to enable or disable an existing administrator.

You can specify the administrators to modify in a number of ways, by piping in existing objects, by passing existing objects with the InputObject parameter, or by specifying the names or SIDs explicitly.

Examples

EXAMPLE 1

Enable all administrators that are currently disabled.

```
Get-AdminAdministrator -Enabled $false | Set-AdminAdministrator -
    Enabled $true
```

EXAMPLE 2

Enable two specific users specified by name (TestUser1 and TestUser2).

```
Set-AdminAdministrator -Name DOMAIN\TestUser1,DOMAIN\TestUser2 -Enabled
    $true
```

Parameters

-InputObject

Specifies the administrators to modify.

Type:	Administrator[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the names of the administrators to modify.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Sid

Specifies the SIDs of the administrators to modify.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Enabled

Specifies the new value for the Enabled property.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Administrator

You can pipe the adminstrators to be modified into this command.

Outputs

None or Citrix.DelegatedAdmin.Sdk.Administrator

When you use the PassThru parameter, Set-AdminAdministrator generates a Citrix.DelegatedAdmin.Sdk.Administra object. Otherwise, this cmdlet does not generate any output.

Related Links

- New-AdminAdministrator
- Get-AdminAdministrator
- Remove-AdminAdministrator
- Set-AdminAdministratorMetadata
- Remove-AdminAdministratorMetadata

Set-AdminAdministratorMetadata

March 11, 2024

Adds or updates metadata on the given Administrator.

Syntax

```
Set-AdminAdministratorMetadata
   -AdministratorSid <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministratorMetadata
   -AdministratorSid <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministratorMetadata
   [-AdministratorName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministratorMetadata
   [-AdministratorName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministratorMetadata
   [-InputObject] <Administrator[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminAdministratorMetadata
   [-InputObject] <Administrator[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Administrator objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Administrator with the identifier ‘S-1-5-21-1505241163-3345470479-1241728991-1000’.

```
Set-AdminAdministratorMetadata -AdministratorSid S
    -1-5-21-1505241163-3345470479-1241728991-1000 -Name property -Value
    value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-AdministratorName

Name of the Administrator

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Administrator[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-AdministratorSid

Sid of the Administrator

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Administrator specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AdminAdministratorMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Remove-AdminAdministratorMetadata

Set-AdminDBConnection

March 11, 2024

Configures a database connection for the DelegatedAdmin Service.

Syntax

```
Set-AdminDBConnection
    [-DBConnection] <String>
    [-Force]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix DelegatedAdmin Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-AdminDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```


EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
1  Set-AdminDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the DelegatedAdmin Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-AdminDBConnection cmdlet returns an object describing the status of the DelegatedAdmin Service together with extra diagnostics information. Possible values are:

- OK:

The DelegatedAdmin Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the DelegatedAdmin Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the DelegatedAdmin Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the DelegatedAdmin Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the DelegatedAdmin Service currently in use is newer than, and incompatible with, the version of the DelegatedAdmin Service schema on the database. Upgrade the DelegatedAdmin Service to a more recent version.

- DBOlderVersionThanService:

The version of the DelegatedAdmin Service schema on the database is newer than, and incompatible with, the version of the DelegatedAdmin Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the DelegatedAdmin Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the DelegatedAdmin Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the DelegatedAdmin Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminServiceStatus
- Get-AdminDBConnection
- Test-AdminDBConnection

Set-AdminDBCredentials

March 11, 2024

Configures the database server SQL credentials for the DelegatedAdmin Service.

Syntax

```
Set-AdminDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix DelegatedAdmin Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional DelegatedAdmin Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-AdminDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-AdminDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-AdminDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-AdminDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-AdminDBSchema
- Set-AdminDBConnection
- Get-Credential

Set-AdminRole

March 11, 2024

Set the properties of a role.

Syntax

```
Set-AdminRole
    [-InputObject] <Role[]>
    [-CanLaunchManage <Boolean>]
    [-CanLaunchMonitor <Boolean>]
    [-Description <String>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminRole
    [-Id] <Guid[]>
    [-CanLaunchManage <Boolean>]
    [-CanLaunchMonitor <Boolean>]
    [-Description <String>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminRole
    [-Name] <String[]>
    [-CanLaunchManage <Boolean>]
    [-CanLaunchMonitor <Boolean>]
    [-Description <String>]
    [-PassThru]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The Set-AdminRole command allows the description of custom roles to be updated. You cannot modify built-in roles.

To modify the permissions of a role, use the Add-AdminPermission and Remove-AdminPermission cmdlets.

To update the metadata associated with a role, use the Set-AdminRoleMetadata and Remove-AdminRoleMetadata cmdlets.

Examples

EXAMPLE 1

Change the description of the 'Supervisor'role.

```
1  Set-AdminRole -Name Supervisor -Description "Helpdesk supervisor role"
```

Parameters

-InputObject

Specifies the roles to update (by object).

Type:	Role[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the roles to update (by id).

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the roles to update (by name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-CanLaunchManage

Supplies the new CanLaunchManage value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CanLaunchMonitor

Supplies the new CanLaunchMonitor value.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Supplies the new description value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Role

You can pipe the roles to be updated into this command.

Outputs

None or Citrix.DelegatedAdmin.Sdk.Role

When you use the PassThru parameter, Set-AdminRole generates a Citrix.DelegatedAdmin.Sdk.Role object. Otherwise, this cmdlet does not generate any output.

Related Links

- New-AdminRole
- Get-AdminRole
- Remove-AdminRole
- Rename-AdminRole
- Set-AdminRoleMetadata
- Remove-AdminRoleMetadata
- Add-AdminPermission
- Remove-AdminPermission

Set-AdminRoleMetadata

March 11, 2024

Adds or updates metadata on the given Role.

Syntax

```
Set-AdminRoleMetadata
   [-RoleId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminRoleMetadata
   [-RoleId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminRoleMetadata
   [-RoleName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminRoleMetadata
   [-RoleName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminRoleMetadata
   [-InputObject] <Role[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminRoleMetadata
   [-InputObject] <Role[]>
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Role objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the Role with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-AdminRoleMetadata -RoleId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -
    Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-RoleId

Id of the Role

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-RoleName

Name of the Role

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Role[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Role specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AdminRoleMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Remove-AdminRoleMetadata

Set-AdminScope

March 11, 2024

Set the properties of a scope.

Syntax

```
Set-AdminScope
   [-InputObject] <Scope[]>
   [-Description <String>]
   [-TenantName <String>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScope
   [-Id] <Guid[]>
   [-Description <String>]
   [-TenantName <String>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScope
   [-Name] <String[]>
   [-Description <String>]
   [-TenantName <String>]
   [-PassThru]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Set-AdminScope command allows the description of scopes and the tenant name to be updated. You cannot modify the built-in 'All' scope.

To change the contents of a scope, use the Add-<Noun>Scope and Remove-<Noun>Scope cmdlets from the corresponding PowerShell snap-in.

To update the metadata associated with a scope, use the Set-AdminScopeMetadata and Remove-AdminScopeMetadata cmdlets.

Examples

EXAMPLE 1

Change the description of the ‘Sales’scope.

```
Set-AdminScope -Name Sales -Description "Sales department desktops"
```

Parameters

-InputObject

Specifies the scopes to update (by object).

Type:	Scope[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

Specifies the scopes to update (by id).

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Name

Specifies the scopes to update (by name).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Description

Supplies the new description value.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenantName

Specifies the new tenant name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Scope

You can pipe the scopes to be updated into this command.

Outputs

None or Citrix.DelegatedAdmin.Sdk.Scope

When you use the PassThru parameter, Set-AdminScope generates a Citrix.DelegatedAdmin.Sdk.Scope object. Otherwise, this cmdlet does not generate any output.

Related Links

- New-AdminScope
- Get-AdminScope
- Remove-AdminScope
- Rename-AdminScope
- Set-AdminScopeMetadata
- Remove-AdminScopeMetadata

Set-AdminScopeMetadata

March 11, 2024

Adds or updates metadata on the given Scope.

Syntax

```
Set-AdminScopeMetadata
   [-ScopeId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-AdminScopeMetadata
   [-ScopeId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScopeMetadata
   [-ScopeName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScopeMetadata
   [-ScopeName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScopeMetadata
   [-InputObject] <Scope[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-AdminScopeMetadata
   [-InputObject] <Scope[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Scope objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Scope with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.


```
Set-AdminScopeMetadata -ScopeId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -
    Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-ScopeId

Id of the Scope

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ScopeName

Name of the Scope

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	Scope[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Scope specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AdminScopeMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Remove-AdminScopeMetadata

Set-AdminServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-AdminServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminServiceMetadata
    [-ServiceHostId] <Guid>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminServiceMetadata
    [-InputObject] <Service[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-AdminServiceMetadata
    [-InputObject] <Service[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-AdminServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```


Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-AdminServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Remove-AdminServiceMetadata

Test-AdminAccess

March 11, 2024

Retrieves the scopes where the specified operation is permitted.

Syntax

```
Test-AdminAccess
    [-Operation] <String[]>
    [-Annotate]
    [-RejectedScopes]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

This cmdlet evaluates what rights the current user has, and from these determines the scopes where the specified operation is permitted.

Operations are the indivisible unit of functionality that each XenDesktop service can perform, and usually correspond to individual cmdlets.

If you specify the -Annotate option or specify multiple operations to check, the resulting object is annotated with the operation the result relates to.

Examples

EXAMPLE 1

Queries the scopes where 'Broker:GetCatalog'is permitted.

```
Test-AdminAccess -Operation 'Broker:GetCatalog'
```

EXAMPLE 2

Queries the scopes where 'Broker:GetCatalog'or 'Broker:GetMachine'are permitted.

```
Test-AdminAccess -Operation 'Broker:GetCatalog','Broker:GetMachine'
```


Parameters

-Operation

The operation to query.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Annotate

Annotates each result with the operation it relates to.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RejectedScopes

Gets the scopes where operations are not allowed as well as the scopes where an operation is allowed

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.DelegatedAdmin.Sdk.ScopeReference

The list of permissible scopes for the specified single operation.

PSObject

The list of permissible scopes for each operation. This type of object is returned when the -Annotate option or multiple operations are specified.

Notes

If the specified operation has unrestricted access a single object is returned representing the 'All' scope with a ScopeId of Guid.Empty (00000000-0000-0000-0000-000000000000).

Related Links

- about_AdminDelegatedAdminSnapin

Test-AdminDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix DelegatedAdmin Service.

Syntax

```
Test-AdminDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AdminDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-AdminDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix DelegatedAdmin Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a DelegatedAdmin SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1  Test-AdminDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
       Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix DelegatedAdmin Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-AdminDBConnection cmdlet returns an object describing the status of the selected DelegatedAdmin Service instance that would result if the connection string were used with the Set-

AdminDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-AdminDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the DelegatedAdmin Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the DelegatedAdmin Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The DelegatedAdmin Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The DelegatedAdmin Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the DelegatedAdmin Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the DelegatedAdmin Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_AdminDelegatedAdminSnapin
- Get-AdminServiceStatus
- Get-AdminDBConnection
- Set-AdminDBConnection

Update-AdminNameCache

March 11, 2024

Updates admin name cache

Syntax

```
Update-AdminNameCache
       [[-Name] <String>]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This command marks the admin names that need to be resynced during the admin sync site service by setting their CloudNameExpiryTime

Examples

EXAMPLE 1

Marks the admin names ending with the letter s

```
Update-AdminNameCache -Name *s
```

Parameters

-Name

Specifies the wild card entry for admin names to be invalidated

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.DelegatedAdmin.Sdk.Name

Takes wild card entry of admin names as input

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_AdminDelegatedAdminSnapin
- Update-AdminNameCache -Name *s

about_EnvTestEnvTestSnapIn

March 11, 2024

Topic

about_EnvTestEnvTestSnapin

Short Description

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation.

Command Prefix

All commands in this snap-in have the noun prefixed with 'EnvTest'.

Long Description

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation at different points during and after configuration and install.

about_EnvTestEnvTestSnapIn

March 11, 2024

Topic

about_EnvTestEnvTestSnapin

Short Description

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation.

Command Prefix

All commands in this snap-in have the noun prefixed with 'EnvTest'.

Long Description

The Citrix Environment Test Service provides tools to test and inspect the state of a XenDesktop installation at different points during and after configuration and install.

about_EnvTest_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
1 # ReturnTotalRecord example
2 Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```



```
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces

quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:


```
Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
```


Get-<Noun> -Filter { StartTime -ge '-0:0:30'} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*"}
Get-<Noun> -Filter { Users -contains "Fred*"}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Get-EnvTestConfiguration

March 11, 2024

Gets the Environment Test Service’s configuation options

Syntax

```
Get-EnvTestConfiguration
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the Environment Test Service's configuation options and returns them as key/value pairs.

Examples

EXAMPLE 1

Gets all configuration options

```
Get-EnvTestConfiguration
```

Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Dictionary<string, object>

All configuration settings

Related Links

Get-EnvTestDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the EnvTest Service.

Syntax

```
Get-EnvTestDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets the database connection string from the currently selected EnvTest Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the EnvTest Service instance running on controller "controller1.mydomain.net".

```
1  Get-EnvTestDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current EnvTest Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the EnvTestService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Set-EnvTestDBConnection
- Get-EnvTestServiceStatus
- Test-EnvTestDBConnection

Get-EnvTestDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix EnvTest Service.

Syntax

```
Get-EnvTestDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix EnvTest Service database schema, add a new EnvTest service to an existing site, remove a EnvTest service from a site, or create a database server logon for a EnvTest service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected EnvTest service instance, otherwise the scripts relate to EnvTest service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to EnvTest service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user

- Addition of database user to EnvTest service roles

If ScriptType is Evict, the returned script contains:

- Removal of EnvTest service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix EnvTest Service and copies it to a file called "C:\EnvTestSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a EnvTest service schema.

```
Get-EnvTestDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\EnvTestSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the EnvTest service. This can be used when configuring a mirror server for use.

```
Get-EnvTestDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    EnvTestLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new EnvTest service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the EnvTest services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix EnvTest Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a EnvTest Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a EnvTest Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the EnvTest Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for EnvTest services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the EnvTest Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Set-EnvTestDBConnection
- Test-EnvTestDBConnection

Get-EnvTestDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix EnvTest Service.

Syntax

```
Get-EnvTestDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets an SQL script that can be used to update the current Citrix EnvTest Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix EnvTest Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix EnvTest Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix EnvTest Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix EnvTest Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-EnvTestServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
1 $update = Get-EnvTestDBVersionChangeScript -DatabaseName MyDb -
     TargetVersion 7.40.0.0
2 $update.Script > update_740.sql
```


Parameters

-DatabaseName

The name of the database containing the Citrix EnvTest Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-EnvTestDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix EnvTest Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any EnvTest services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-EnvTestServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestInstalledDBVersion
- Get-EnvTestServiceStatus
- Get-EnvTestDBSchema

Get-EnvTestDefinition

March 11, 2024

Gets the one or more test definitions

Syntax

```
Get-EnvTestDefinition
   [-TestId <String[]>]
   [-CultureName <String>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns a list of test definitions that are available from currently running components.

Examples

EXAMPLE 1

Retrieve all tests.

```
$allTestDefinitions = Get-EnvTestDefinition
```

EXAMPLE 2

Retrieve all tests with localized properties returned in Spanish.

```
$allTestDefinitionsTranslatedIntoSpanish = Get-EnvTestDefinition -
    CultureName es-ES
```

EXAMPLE 3

Retrieve the definition of the 'Monitor_RegisteredWithConfigurationService'test.

```
$monitorConfigServiceRegistrationDefinition = Get-EnvTestDefinition -
    TestId Monitor_RegisteredWithConfigurationService
```

Parameters

-TestId

The id of one or more tests.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-CultureName

The culture name in which to produce results. The culture name is in standard language/region-code format; for example "en-US".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

A test id.

String[]

An array of test ids.

Outputs

Citrix.EnvTest.Sdk.EnvTestDefinition

One or more test definitions.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestSuiteDefinition
- Get-EnvTestTask
- Start-EnvTestTask
- Switch-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Get-EnvTestInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the EnvTest Service.

Syntax

```
Get-EnvTestInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix EnvTest Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-EnvTestDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix EnvTest Service database schema version number.

```
Get-EnvTestInstalledDBVersion
```

EXAMPLE 2

Get the versions of the EnvTest Service database schema for which upgrade scripts are supplied.

```
Get-EnvTestInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-EnvTestInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the EnvTest

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDBVersionChangeScript
- Get-EnvTestDBSchema

Get-EnvTestService

March 11, 2024

Gets the service record entries for the EnvTest Service.

Syntax

```
Get-EnvTestService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the EnvTest Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the EnvTest Service running in the current service group.

```
Get-EnvTestService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```


Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*"matches records with a metadata entry having a key name of "abc"and a value starting with the letter "x".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_EnvTest_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_EnvTest_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.EnvTest.Sdk.Service

The Get-EnvTestServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin

Get-EnvTestServiceAddedCapability

March 11, 2024

Gets any added capabilities for the EnvTest Service on the controller.

Syntax

```
Get-EnvTestServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the EnvTest Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the EnvTest Service.

```
Get-EnvTestServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin

Get-EnvTestServiceInstance

March 11, 2024

Gets the service instance entries for the EnvTest Service.

Syntax

```
Get-EnvTestServiceInstance
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the EnvTest Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the EnvTest Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-EnvTestServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.EnvTest.Sdk.ServiceInstance

The Get-EnvTestServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always EnvTest.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.EnvTest.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestServiceStatus
- Reset-EnvTestServiceGroupMembership

Get-EnvTestServiceStatus

March 11, 2024

Gets the current state of the EnvTest Service on the controller.

Syntax

```
Get-EnvTestServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the EnvTest Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return

DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the EnvTest Service.

```
Get-EnvTestServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-EnvTestServiceStatus command returns an object containing the status of the EnvTest Service together with extra diagnostics information.

DBUnconfigured

The EnvTest Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the EnvTest Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the EnvTest Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The EnvTest is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The EnvTest is connected to a database that is valid, but it does not have the full functionality required so the EnvTest cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the EnvTest Service currently in use is incompatible with the version of the EnvTest Service schema on the database. Upgrade the EnvTest Service to a more recent version.

DBOlderVersionThanService

The version of the EnvTest Service schema on the database is incompatible with the version of the EnvTest Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The EnvTest Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the EnvTest Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the EnvTest and the database has been lost for an extended period of time, or has failed due to a configuration problem. The EnvTest service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Set-EnvTestDBConnection
- Test-EnvTestDBConnection
- Get-EnvTestDBConnection
- Get-EnvTestDBSchema

Get-EnvTestSuiteDefinition

March 11, 2024

Gets one or more test suite definitions.

Syntax

```
Get-EnvTestSuiteDefinition
   [-TestSuiteId <String[]>]
   [-CultureName <String>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a list of test suite definitions that are available from currently running components.

Examples

EXAMPLE 1

Retrieve all test suites.

```
$allTestSuiteDefinitions = Get-EnvTestSuiteDefinition
```

EXAMPLE 2

Retrieve all test suites with localized properties returned in Spanish.

```
$allTestSuiteDefinitionsTranslatedIntoSpanish = Get-
   EnvTestSuiteDefinition -CultureName es-ES
```

EXAMPLE 3

Retrieve the definition of the 'Infrastructure'test suite.

```
$infrastructureSuiteDefinition = Get-EnvTestSuiteDefinition -
   TestSuiteId Infrastructure
```


Parameters

-TestSuiteId

The id of one or more test suites.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-CultureName

The culture name in which to produce results. The culture name is in standard language/region-code format; for example “en-US”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

A test suite id.

String[]

An array of test suite ids.

Outputs

Citrix.EnvTest.Sdk.EnvTestSuiteDefinition

The definition of a test suite

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestTask
- Start-EnvTestTask
- Switch-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Get-EnvTestTask

March 11, 2024

Gets one or more EnvTestTask(s)

Syntax


```
Get-EnvTestTask
    [-TaskId <Guid>]
    [-List]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns either the current task, a specified task, or list of tasks that are currently known to the EnvTest Service.

Examples

EXAMPLE 1

Retrieve the current task. The current task is the most recently created task unless Switch-EnvTestTask explicitly changes it.

```
$currentTask = Get-EnvTestTask
```

EXAMPLE 2

Retrieve a fresh copy of a task object based on a known task id, which is always a Guid. This id can be retrieved from an existing task object via its $task.TaskId property.

```
$taskOfSpecificId = Get-EnvTestTask -TaskId 36C0EC52-2039-4D6E-B690-9
    F02F8CEFFCC
```

EXAMPLE 3

Retrieve the list of current tasks. This list includes any task started by the Start-EnvTestTask cmdlet since the service started that has not later been removed via Remove-EnvTestTask.

```
$allKnownTasks = Get-EnvTestTask -List
```

Parameters

-TaskId

Specifies the task identifier to be returned. This value can be retrieved from an existing task's $task.TaskId property.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-List

List all running tasks, including the current one.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.EnvTest.Sdk.EnvTestTask

A description of a previously started task.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition
- Start-EnvTestTask
- Switch-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

New-EnvTestDiscoveryTargetDefinition

March 11, 2024

Creates a new EnvTestDiscoveryTargetDefinition object

Syntax

```
New-EnvTestDiscoveryTargetDefinition
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-EnvTestDiscoveryTargetDefinition
   -TestId <String>
   [-TargetIdType <String>]
   [-TargetId <String>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-EnvTestDiscoveryTargetDefinition
   -TestSuiteId <String>
   [-TargetIdType <String>]
   [-TargetId <String>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Creates a new EnvTestDiscoveryTargetDefinition object that can be piped into Start-EnvTestTask to define one or more targets of execution, optionally including root objects for discovery.

Examples

EXAMPLE 1

Create a discovery target definition with a single test and no target object, then start a task based on it.

```
$singleSimpleTestTaskTarget = New-EnvTestDiscoveryTargetDefinition -
    TestId Monitor_RegisteredWithConfigurationService
$singleSimpleTestTaskTarget | Start-EnvTestTask
```

EXAMPLE 2

Create a discovery target definition with a single test suite and no target object, then start a task based on it.

```
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition
     -TestSuiteId Infrastructure
$singleSimpleTestSuiteTaskTarget | Start-EnvTestTask
```

EXAMPLE 3

Create a discovery target definition with a single test suite and a catalog target object, then start a task based on it.

```
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -
    TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-
    BrokerCatalog).Uuid
$singleTestSuiteTaskTarget | Start-EnvTestTask
```

EXAMPLE 4

Create two different discovery target definitions, put them in an array, then start a task based on both.


```
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition
    -TestSuiteId Infrastructure
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -
   TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-
   BrokerCatalog).Uuid
@($singleSimpleTestSuiteTaskTarget, $singleTestSuiteTaskTarget) | Start
   -EnvTestTask
```

Parameters

-TestId

Test identifiers. If specified, do not specify -TestSuiteId.

Type:	String
Position:	Named
Default value:	Empty
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TestSuiteId

Test suite identifiers. If specified, do not specify -TestId.

Type:	String
Position:	Named
Default value:	Empty
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetIdType

Describes the type of corresponding object passed with -TargetId

Type:	String
Position:	Named
Default value:	Empty
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetId

The Ids that object tests or test suites will target. By default, other components are queried for objects related to these.

Type:	String
Position:	Named
Default value:	Empty
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition

Defines a target of a task

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition
- Get-EnvTestTask
- Start-EnvTestTask
- Switch-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Remove-EnvTestServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-EnvTestServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-EnvTestServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-EnvTestService | % {
  Remove-EnvTestServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Set-EnvTestServiceMetadata

Remove-EnvTestTask

March 11, 2024

Removes from the database completed tasks for the EnvTest Service.

Syntax

```
Remove-EnvTestTask
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestTask
      [-TaskId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestTask
      [-Task <EnvTestTask>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables completed tasks that have run within the EnvTest Service to be removed from the database.

Examples

EXAMPLE 1

Removes the current task from the EnvTest service.

```
Remove-EnvTestTask
```

EXAMPLE 2

Removes a task from the EnvTest service via its id, which is available from an existing task's $task.TaskId property.

```
Remove-EnvTestTask -TaskId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
```

EXAMPLE 3

Removes the second task in the list returned by Get-EnvTestTask -List.

```
$secondTask = $(Get-EnvTestTask -List)[1]
Remove-EnvTestTask -Task $secondTask
```

Parameters

-TaskId

Specifies the identifier of the task to be removed, retrieveable from the $task.TaskId property.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Task

Specifies the task to be removed.

Type:	EnvTestTask
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition

- Get-EnvTestTask
- Start-EnvTestTask
- Switch-EnvTestTask
- Stop-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Remove-EnvTestTaskMetadata

March 11, 2024

Removes metadata from the given Task.

Syntax

```
Remove-EnvTestTaskMetadata
      [-TaskTaskId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestTaskMetadata
      [-TaskTaskId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestTaskMetadata
      [-InputObject] <EnvTestTask[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-EnvTestTaskMetadata
      [-InputObject] <EnvTestTask[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to remove metadata from the given Task.

Examples

EXAMPLE 1

Remove all metadata from all Task objects.

```
Get-EnvTestTask | % {
  Remove-EnvTestTaskMetadata -Map $_.MetadataMap  }
```

Parameters

-TaskTaskId

Id of the Task

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	EnvTestTask[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Set-EnvTestTaskMetadata

Reset-EnvTestEnabledFeatureList

March 11, 2024

Refreshes the EnvTest service's list of enabled features.

Syntax

```
Reset-EnvTestEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix EnvTest Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected EnvTest service instance's list of enabled features.

```
Reset-EnvTestEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-EnvTestServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the EnvTest Service.

Syntax

```
Reset-EnvTestServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload EnvTest Service access permissions and configuration service locations. The Reset-EnvTestServiceGroupMembership command must be run on at least one instance of the service type (EnvTest) after installation and registration with the configuration service. Without this operation, the EnvTest services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-EnvTestServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    EnvTestServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-EnvTestServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.EnvTest.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-EnvTestServiceGroupMembership command.

Outputs

Citrix.EnvTest.Sdk.ServiceInstance

Reset-EnvTestServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the EnvTest Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestServiceInstance
- Get-EnvTestServiceStatus

Set-EnvTestConfiguration

March 11, 2024

Sets the Environment Test Service's configuation options

Syntax

```
Set-EnvTestConfiguration
   [-PerTypeDiscoveredObjectLimit <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Sets the Environment Test Service's configuation options and broadcasts the changes to other machines in the Site.

Examples

EXAMPLE 1

Set the maximum to 100

```
Set-EnvTestConfiguration -PerTypeDiscoveredObjectLimit 100
```

Parameters

-PerTypeDiscoveredObjectLimit

Sets the maximum number of objects of a particular type to be explored when discovering objects during a test run. Default: 50

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Integer

Must be greater than zero.

Outputs

String

Related Links

Set-EnvTestDBConnection

March 11, 2024

Configures a database connection for the EnvTest Service.

Syntax

```
Set-EnvTestDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix EnvTest Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-EnvTestDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-EnvTestDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the EnvTest Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-EnvTestDBConnection cmdlet returns an object describing the status of the EnvTest Service together with extra diagnostics information. Possible values are:

- OK:

The EnvTest Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the EnvTest Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the EnvTest Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the EnvTest Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the EnvTest Service currently in use is newer than, and incompatible with, the version of the EnvTest Service schema on the database. Upgrade the EnvTest Service to a more recent version.

- DBOlderVersionThanService:

The version of the EnvTest Service schema on the database is newer than, and incompatible with, the version of the EnvTest Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the EnvTest Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the EnvTest Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the EnvTest Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestServiceStatus
- Get-EnvTestDBConnection
- Test-EnvTestDBConnection

Set-EnvTestDBCredentials

March 11, 2024

Configures the database server SQL credentials for the EnvTest Service.

Syntax

```
Set-EnvTestDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-EnvTestDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix EnvTest Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional EnvTest Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-EnvTestDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-EnvTestDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-EnvTestDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-EnvTestDBCredentials $null
```


Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-EnvTestDBSchema
- Set-EnvTestDBConnection
- Get-Credential

Set-EnvTestServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-EnvTestServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-EnvTestServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-EnvTestServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-EnvTestServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-EnvTestServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                          Value
---                                          -----
property                                     value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-EnvTestServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Remove-EnvTestServiceMetadata

Set-EnvTestTaskMetadata

March 11, 2024

Adds or updates metadata on the given Task.

Syntax

```
Set-EnvTestTaskMetadata
   [-TaskTaskId] <Guid>
   -Name <String>
   -Value <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-EnvTestTaskMetadata
    [-TaskTaskId] <Guid>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-EnvTestTaskMetadata
    [-InputObject] <EnvTestTask[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-EnvTestTaskMetadata
    [-InputObject] <EnvTestTask[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Task objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-EnvTestTaskMetadata -TaskTaskId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```


Parameters

-TaskTaskId

Id of the Task

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	EnvTestTask[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-EnvTestTaskMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Remove-EnvTestTaskMetadata

Start-EnvTestTask

March 11, 2024

Starts a new test task.

Syntax

```
Start-EnvTestTask
     [-IgnoreRelatedObjects]
     [-RunAsynchronously]
     [-ExcludeNotRunTests]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Start-EnvTestTask
     -TestId <String>
     [-TargetIdType <String>]
     [-TargetId <String>]
     [-CultureName <String>]
     [-IgnoreRelatedObjects]
     [-RunAsynchronously]
     [-ExcludeNotRunTests]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Start-EnvTestTask
     -TestSuiteId <String>
     [-TargetIdType <String>]
     [-TargetId <String>]
     [-CultureName <String>]
     [-IgnoreRelatedObjects]
     [-RunAsynchronously]
     [-ExcludeNotRunTests]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Start-EnvTestTask
     [-CultureName <String>]
     -InputObject <PSObject[]>
     [-IgnoreRelatedObjects]
     [-RunAsynchronously]
     [-ExcludeNotRunTests]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```


Description

Starts a new test task based on a set of criteria provided via parameters or piped input and either waits for the tests to run or returns immediately depending on how it is called. When running a test suite and providing a target object for that suite, the service will discover related objects by default, but this behavior may be disabled if desired.

Examples

EXAMPLE 1

Create and start a task with a single test and no target object.

```
$singleSimpleTestTask = Start-EnvTestTask -TestId
    Monitor_RegisteredWithConfigurationService
```

EXAMPLE 2

Create and start a task with a single test and no target object, with localized properties translated into Spanish.

```
$singleSimpleTestTaskInSpanish = Start-EnvTestTask -TestId
    Monitor_RegisteredWithConfigurationService -CultureName es-ES
```

EXAMPLE 3

Create and start a task with a single test suite and no target object.

```
$singleSimpleTestSuiteTask = Start-EnvTestTask -TestSuiteId
    Infrastructure
```

EXAMPLE 4

Create and start a task with a single test suite and a catalog target object.

```
$singleTestSuiteTask = Start-EnvTestTask -TestSuiteId Catalog -
    TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid
```

EXAMPLE 5

Create and start a task with a single test suite and a catalog target object, and return immediately not waiting for the tests to complete.


```
$singleTestSuiteTask = Start-EnvTestTask -TestSuiteId Catalog -
    TargetIdType Catalog -TargetId $(Get-BrokerCatalog).Uuid -
    RunAsynchronously
```

EXAMPLE 6

Create and start a task with a single test, a target object for that test, and no object discovery based on that target.

```
$adAccountPool = Get-AcctIdentityPool
$singleTestTaskWithNoObjectDiscovery = StartEnvTestTask -
    IgnoreRelatedObjects -TestId
    ADIdentity_IdentityPool_ValidatePoolIsUnlocked -TargetIdType
    IdentityPool -TargetId $adAccountPool.IdentityPoolUid
```

EXAMPLE 7

Create two different discovery target definitions, put them in an array, then start a task based on both via -InputObject.

```
$singleSimpleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition
     -TestSuiteId Infrastructure
$singleTestSuiteTaskTarget = New-EnvTestDiscoveryTargetDefinition -
    TestSuiteId Catalog -TargetIdType Catalog -TargetId $(Get-
    BrokerCatalog).Uuid
$inputObjects = @($singleSimpleTestSuiteTaskTarget,
    $singleTestSuiteTaskTarget)
Start-EnvTestTask -InputObject $inputObjects
```

Parameters

-TestId

Test identifiers. If specified, do not specify -TestSuiteId.

Type:	String
Position:	Named
Default value:	Empty
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-TestSuiteId

Test suite identifiers. If specified, do not specify -TestId.

Type:	String
Position:	Named
Default value:	Empty
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

One or more test targets defining the task, see Input Types for details about what kind of objects are permissible. Any valid object passed to this parameter may also be piped into this cmdlet.

Type:	PSObject[]
Position:	Named
Default value:	Empty
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-TargetIdType

Describes the type of corresponding object passed with -TargetId

Type:	String
Position:	Named
Default value:	Empty

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetId

The Ids that object tests or tests suites will target. By default, other components are queried for objects related to these.

Type:	String
Position:	Named
Default value:	Empty
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CultureName

The culture name in which to produce results. The culture name is in standard language/region-code format; for example “en-US”.

Type:	String
Position:	Named
Default value:	The current user interface culture
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IgnoreRelatedObjects

Do not ask other components for objects related to a specified target.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Do not wait for the test run to complete, return immediately.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludeNotRunTests

If set, tests that are not run because no object matching their requirements is found are NOT included in test counts and results.

Type:	SwitchParameter
Position:	Named
Default value:	False (include Not Run tests)
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition

A single EnvTestDiscoveryTargetDefinition can be specified to target one test or test suite.

Citrix.EnvTest.Sdk.EnvTestDiscoveryTargetDefinition[]

An array of EnvTestDiscoveryTargetDefinition(s) can be specified to target any combination of tests and/or test suites.

PSCustomObject

A single PSCustomObject with properties matching the required EnvTestDiscoveryTargetDefinition properties can be specified to target one test or test suite.

PSCustomObject[]

An array of PSCustomObject(s) with properties matching the required EnvTestDiscoveryTargetDefinition properties can be specified to target any combination of tests and/or test suites.

Outputs

Ctrix.EnvTest.Sdk.EnvTestTask

The newly started task.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition
- Get-EnvTestTask
- New-EnvTestDiscoveryTargetDefinition
- Switch-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Stop-EnvTestTask

March 11, 2024

Stops a still running task from completing.

Syntax

```
Stop-EnvTestTask
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Stop-EnvTestTask
    [-TaskId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Stop-EnvTestTask
    [-Task <EnvTestTask>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Stops a still running task from completing. A task may still be retreived via Get-EnvTestTask until it Remove-EnvTestTask is called with its task id.

Examples

EXAMPLE 1

Stops the current task from completing if it is still running.

```
Stop-EnvTestTask
```

EXAMPLE 2

Stops a task from completing via its id, which is available from an existing task's $task.TaskId property.

```
Stop-EnvTestTask -TestId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
```

EXAMPLE 3

Stops the second task in the list returned by Get-EnvTestTask -List.

```
$secondTask = $(Get-EnvTestTask -List)[1]
Stop-EnvTestTask -Task $secondTask
```

Parameters

-TaskId

The id of the task to stop, retrieveable from the $task.TaskId property.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Task

An EnvTestTask representing the task to stop

Type:	EnvTestTask
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition

- Get-EnvTestTask
- Start-EnvTestTask
- Switch-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Switch-EnvTestTask

March 11, 2024

Sets the current task that will be returned by a call to Get-EnvTestTask with no parameters.

Syntax

```
Switch-EnvTestTask
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Switch-EnvTestTask
      [-TaskId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Switch-EnvTestTask
      [-Task <EnvTestTask>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Sets the current task that will be returned by a call to Get-EnvTestTask with no parameters.

Examples

EXAMPLE 1

Switches the current task to another via its id, which is available from an existing task's $task.TaskId property.

```
Switch-EnvTestTask -TaskId 50A4139F-2B55-4A97-A1BE-20EE4E124AA3
```


EXAMPLE 2

Switches the current task to the second in the list returned by Get-EnvTestTask -List.

```
$secondTask = $(Get-EnvTestTask -List)[1]
Switch-EnvTestTask -Task $switchTask
```

Parameters

-TaskId

Specifies the identifier of the task to be made current.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Task

The task object to be made current, retrieveable from the $task.TaskId property.

Type:	EnvTestTask
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Remove-EnvTestTask command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestDefinition
- Get-EnvTestSuiteDefinition
- Get-EnvTestTask
- Start-EnvTestTask
- Stop-EnvTestTask
- Remove-EnvTestTask
- Set-EnvTestTaskMetadata
- Remove-EnvTestTaskMetadata

Test-EnvTestDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix EnvTest Service.

Syntax

```
Test-EnvTestDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-EnvTestDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-EnvTestDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix EnvTest Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a EnvTest SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.


```
Test-EnvTestDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix EnvTest Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-EnvTestDBConnection cmdlet returns an object describing the status of the selected EnvTest Service instance that would result if the connection string were used with the Set-EnvTestDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-EnvTestDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the EnvTest Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the EnvTest Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The EnvTest Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The EnvTest Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the EnvTest Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the EnvTest Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_EnvTestEnvTestSnapin
- Get-EnvTestServiceStatus
- Get-EnvTestDBConnection
- Set-EnvTestDBConnection

about_HypHostSnapIn

March 11, 2024

Topic

about_HypHostSnapin

Short Description

The Host Service PowerShell snap-in provides administrative functions for the Host Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Hyp'.

Long Description

The Host Service PowerShell snap-in enables both local and remote administration of the Host Service. It lets you configure XenDesktop deployments to make use of hypervisors, networks, and storage, and enables browsing of their contents using the Host PowerShell provider (Citrix.HypervisorProvider).

The provider creates a default PSDrive with a drive identifier of 'XDHyp'. This drive provides two root folders:

HostingUnits Folder

Contains a list of all the hosting units that have been defined using the new-Item provider command.

Hosting units define not only a hypervisor connection, but also reference networks and storage. Hosting units are used by the Machine Creation Service to provide the information required to create and manage virtual machines that can be used by other services. A hosting unit references a root path. This is a specific point in the provider connection tree. The hosting unit is constrained to provide only items below this point in the tree. This restricts the locations that the Machine Creation Service can use to create virtual machines and the networks and storage that can be used.

Connections Folder

Contains a list of all the hosting unit connections that are defined using the new-Item provider command.

Change directory to a specific connection and use the dir/Get-ChildItem command to list all of the infrastructure (such as folders, hypervisors, networks, storage, and virtual machines) that is available in the hosting unit to which the connection refers. The paths to these items are used as the input to commands in the Host Service and Machine Creation Service snap-ins.

The contents of the Hosting Unit and Connection folders reflect the content and structure of the hypervisor to which they refer. The item extensions

reflect the object type for each item. Not all item types are appropriate for all hypervisor types. Possible item types are:

Virtual Machine (.vm)
Snapshot (.snapshot)
Cluster (.cluster)
Host (.host)
HostGroup (.hostgroup)
DataCenter (.datacenter)
Folder (.folder)
ResourcePool (.resourcepool)
ComputeResource (.computeresource)

When used with a path that refers to the Host provider (with a default drive of 'XDHyp:'), the Host provider extends the standard New-Item, Get-Item, Get-ChildItem, Remove-Item, Rename-Item, and Set-Item commands as described below. For more information about the basic behavior of these commands, see the help for the command. The Move-Item and Copy-Item commands are not supported for the Host provider.

New-Item

The following parameters are available when using New-Item in the Connections directory (or a path that refers to the directory). The Credential parameter is not supported.

-Name

Specifies the name of the connection, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the connection name must be specified as part of the Path parameter.

-HypervisorAddress <String[]>

Specifies an array of addresses that can be used to contact the required hypervisor. All the addresses are considered equivalent, that is, all of the addresses provide access to the same virtual machines, snapshots, network, and storage.

-ConnectionType <ConnectionType>

Specifies the type of hypervisor that the connection is for. Supported hypervisor types are:

XenServer
SCVMM (Microsoft Hyper-V)

vCenter (VMware vSphere/ESX)
Custom

-PluginId <String>

Specifies the class name for the Citrix Managed Machine library that is used to access the hypervisor. You can obtain this list using the Get-HypHypervisorPlugin command. The PluginId parameter must be specified if the ConnectionType is set to 'Custom'.

-UserName <String>, -Password <String>, -SecurePassword <SecureString>

Specifies the credentials for the connection to the hypervisor. You can specify the password as either Password or SecurePassword, but not both. The UserName parameter, and either Password or SecurePassword, specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

-HypervisorCredential <PSCredential>

Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as UserName and either Password or SecurePassword, so use of one precludes use of the other.

-Persist

Specifies that the connection details are persistent. If this parameter is not included, the connection is held only for the duration of the current runspace and PSDrive combination. Only persistent connection items are visible to administrators in other runspaces and PSDrives. Hosting units cannot be created from connections that are not persistent.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create

high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-Scopes <String[]>

Specifies the list of administrative scopes that the new connection will be a part of. The scopes control which administrators are able to work with the connection.

-ZoneUid <Guid>

Specifies the ZoneUid the connection belongs to. The zone determines which connections are local to the controller.

-IncludeUnavailable <Boolean>

Specifies whether or not to show all hypervisor plug-ins regardless of availability.

The following parameters are available when using New-Item in the HostingUnits directory (or a path that refers to the directory).

-Name

Specifies the name of the hosting unit, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the hosting unit name must be provided as part of the Path parameter.

-HypervisorConnectionName <String>

Specifies the name of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

-HypervisorConnectionUid <Guid>

Specifies the unique identifier of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

-RootPath <String>

Specifies the location in a connection that is used as the starting reference for the hosting unit. The path must point to an item in the connection that is marked as a SymLink. The root of a XenServer connection is a special case that is also considered a SymLink. If this parameter is not included, the current location in the provider is used.

-NetworkPath <String>

Specifies the path in a connection to the network item that is used when the Machine Creation Service creates new virtual machines.

-StoragePath <String>

Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates new virtual machines. After they are set, you can modify storage paths using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands. If the connection is based on cloud infrastructure, storage items are typically not available, in which case this parameter can be omitted.

-PersonalvDiskStoragePath <String>

Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates disks for the virtual machines. After they are set, you can modify storage paths using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands.

-NoVmTagging

Specifies that new virtual machines are not tagged with metadata from the hypervisor. By default, all virtual machines created by the Machine Creation Service are tagged to show they are created by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>

Specifies the address of the Host Service that the command will communicate with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching

When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available, use this parameter to specify that

the virtual machines created for this hosting unit will use local storage caching for their disk images.

-GpuGroup

Specifies a path to a GPU Group in a connection that will be used when Machine Creation Services creates VMs. Only a single GPU Group is supported and the value is immutable.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-JobGroup <Guid>

Specifies the Uid associuated with the HypStorage object created with the New-HypStorage command. This HypStorage object specifies any additional storage details required for the new connection.

The following parameters are available when using New-Item relative to a connection or hosting unit.

-ItemType <String>

Specifies the type of item to create when invoked relative to a connection or hosting unit. Supported ItemType values are:

SecurityGroup (cloud only)

-Description

Specifies a description for the security group.

-VpcId

Specifies a VPC ID for the security group.

Get-Item

Get-ChildItem (alias dir)

Rename-Item

Remove-Item

The following additional parameters are available.

AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

You can specify the Path parameter by name or by using the unique identifier for the connection or hosting unit enclosed in braces. For example:

-Path XDHyp:{1233-3213ACDF-12323}

The Filter parameter is not supported. Virtual machines created by the Machine Creation Service are returned only if the -Force parameter is used or if virtual machine tagging was disabled on the hosting unit with the NoVmTagging parameter when the VMs were created.

Set-Item

The following parameters are available when using Set-Item in the Connections directory.

-UserName <String>, -Password <String>, -SecurePassword <SecureString>

Specifies the credentials for the connection to the hypervisor. The password can be given as either Password or SecurePassword, but not both. The UserName parameter and either Password or SecurePassword specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

-HypervisorCredential <PSCredential>

Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as the UserName parameter and either Password or SecurePassword, so use of one precludes use of the other.

-HypervisorAddress <String[]>

Specifies the addresses of the hypervisor that this connection represents. This replaces all existing addresses.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-MaintenanceMode <Boolean>

Places the connection into maintenance mode which disables all communication between XenDesktop and the Hypervisor. Use this mode when making changes to the hypervisor; for instance, if the password for the access account needs to be changed.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command tries to use a local Host Service.

The Credential parameter is not supported. Addresses can be modified with the Add-HypHypervisorConnectionAddress and Remove-HypHypervisorConnectionAddress commands. Metadata can be modified with the Add-HypMetadata and Remove-HypMetadata commands.

The following parameters are available when using New-Item in the HostingUnits directory.

-Name

Specifies the name of the hosting unit, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the hosting unit name must be provided as part of the Path parameter.

-NetworkPath <String>

Specifies the path in a connection to the network item that is used when the Machine Creation Service creates new virtual machines.

-NoVmTagging

Specifies that new virtual machines are not tagged with metadata from the hypervisor. By default, all virtual machines created by the Machine Creation Service are tagged to show they are created by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching <Boolean>

When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available, use this parameter to specify that the virtual machines created for this hosting unit will use local storage caching for their disk images.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Storage can be modified with the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands. Metadata can be modified with the Add-HypMetadata and Remove-HypMetadata commands.

The item types returned when these commands are used in the Host Service provider are defined in the Object Definitions section below.

Examples

To Create a Persistent Hypervisor Connection

new-item -path “xdhyp:\Connections”-Name MyConn -ConnectionType


```
XenServer -HypervisorAddress “http:\\address”-UserName user
-Password password -Persist

PSPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\Connections\MyConn

PSParentPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\Connections

PSChildName: MyConn
PSDrive: XDHyp
PSProvider: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor

PSIsContainer: True
HypervisorConnectionUid: 04e6daa2-5cbd-4491-b70c-6daf733ee82a
HypervisorConnectionName: MyConn
ConnectionType: XenServer
HypervisorAddress: {http:\\address}
UserName: user
Persistent: True
PluginId: XenFactory
SupportsPvsVMs: True
Revision: 1b0b0d02-bc1b-49d8-b2b0-be6fb7f150ad
MaintenanceMode: False
Metadata: {MaxAbsoluteActiveActions = 100,

MaxAbsoluteNewActionsPerMinute = 100,
MaxPowerActionsPercentageOfDesktops = 20}

To Create a Hosting Unit

new-item -Path “xdhyp:\HostingUnits”

-Name MyHU
-HypervisorConnectionName MyConn
-RootPath XDHYP:\Connections\MyConn
-NetworkPath XDHYP:\Connections\MyConn\Network 0.network
-StoragePath XDHYP:\Connections\MyConn\Local storage on
myXenServer.storage

PSPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\HostingUnits\MyHU

PSParentPath: Citrix.HostingUnitService.Admin.V1.0\
```


Citrix.Hypervisor::XDHyp:\HostingUnits

PSChildName: MyHU
PSDrive: XDHyp
PSProvider: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor

PSIsContainer: True
HostingUnitUid: df91f886-1141-4280-bd59-2ee260a4df79
HostingUnitName: MyHU
HypervisorConnection: MyConn
RootPath: /
RootId:
NetworkPath: /Network 0.network
NetworkId: ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage: {/Local storage on myXenServer.storage}
VMTaggingEnabled: True
Metadata: {}

Relative paths can be used for all parameters.

Object Definitions

Citrix.XDInterServiceTypes.HypervisorConnection

The hypervisor connection object is returned when a connection item is located. This item has the following parameters.

HypervisorConnectionUid <Guid>

Specifies the unique identifier for the connection item.

HypervisorConnectionName <String>

Specifies the name of the connection item.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>

Specifies the type of hypervisor that the connection is for.
Supported hypervisor types are:

XenServer
SCVMM (Microsoft Hyper-V)
vCenter (VMware vSphere/ESX)
Custom

HypervisorAddress <String[]>

Specifies the addresses that can be used to contact the required hypervisor.

Username <String>

Specifies the administrator user name for the connection to the hypervisor (from the user name and password given as administrator credentials when

setting up this connection)

Persistent <Boolean>

Specifies whether or not the connection is persistent.

PluginId <String>

Specifies the Citrix Managed Machine class identifier for the hypervisor.

SupportsPvsVM <Boolean>

Specifies whether or not the connection can be used as part of a hosting unit. If this parameter is set to 'True', a hosting unit referencing this connection can be created.

Revision <Guid>

A unique identifier that is changed every time any properties of the connection are changed.

MaintenanceMode <Boolean>

Specifies whether or not the connection is currently in maintenance mode.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>

The collection of metadata associated with the connection.

Citrix.XDInterServiceTypes.HostingUnit

The hosting unit object is returned when a hosting unit item is located. This item has the following parameters.

HostingUnitName <String>

Specifies the name of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HypervisorConnection <Citrix.XDInterServiceTypes.HypervisorConnection>

Specifies the hypervisor connection item that the hosting unit references.

NetworkId <String>

Specifies the unique identifier for the network in the hypervisor context that the hosting unit references.

NetworkPath <String>

Specifies the path to the network in the Host Service provider.

RootId <String>

Specifies the unique identifier for the root connection item in the hypervisor context that the hosting unit references.

RootPath <String>

Specifies the path to the root of the hosting unit from within the Host Service provider.

Storage <Citrix.XDInterServiceTypes.Storage[]>

Specifies the collection of storage objects that are defined for use as part of the hosting unit. This object contains the following parameters.

StorageID <String>

Specifies the identifier for the storage in the hypervisor context.

StoragePath <String>

Specifies the path to the storage in the Host Service provider.

VMTaggingEnabled <Boolean>

Specifies whether or not virtual machine metadata is used to tag virtual machines created by XenDesktop.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>

Specifies the collection of metadata associated with the hosting unit.

Citrix.HostingUnitService.Sdk.HypervisorObject

The hypervisor object is returned when any item is located from within a Connection or HostingUnit folder. This item has the following parameters.

AdditionalData <Dictionary<String,String>

Stores extra untyped data for the item. This dictionary contains
different information for each item type.

For Storage Items

Key = StorageType
Values = Shared or Local

For VM Items

Key = PowerState
Values = PoweredOn,

PoweredOff,
Suspended,
TransitioningToOn,
TransitioningToOff,
Suspending,
Resuming,
Unknown,
Error

Key = TemplateIsWindowsTemplate
Values = True or False

For Snapshot Items

Key = TemplateIsWindowsTemplate
Values = True or False

For Azure Managed Disk Items

Key = TemplateIsWindowsTemplate
Values = True or False

For Azure SIG Image Definition Items

Key = TemplateIsWindowsTemplate
Values = True or False

Name <String>

Specifies the name of the item.

FullName <String>

Specifies the name with the appropriate file extension.

ObjectPath <String>

Specifies the relative path to the item from the root of the connection of which the item is part.

FullPath <String>

Specifies the absolute path to the item in the Citrix.Hypervisor provider.

Id <String>

Specifies the identity of the item in the hypervisor context.

IsContainer <Boolean>

Specifies whether or not the item can contain other items.

IsSymLink <Boolean>

Specifies whether or not the item can be used as a RootPath of a hosting unit.

ObjectType <Citrix.HostingUnitService.Sdk.NodeType>

Specifies the item type.

Error Codes

The provider commands can return the following error codes.

New-Item

ConnectionNameOrUidInvalid

The name or unique identifier of the hypervisor connection specified for the hosting unit is invalid.

HostingUnitRootPathInvalid

The root path specified for the hosting unit is invalid. The root path must be a valid item in the hypervisor tree and a SymLink.

HostingUnitNetworkPathInvalid

The network path specified for the hosting unit is invalid. The network path must be a valid item in the hypervisor tree and relative to the root path.

HostingUnitStoragePathInvalid

The storage path specified for the hosting unit is invalid. The storage path must be a valid item in the hypervisor tree and relative to the root path.

HostingUnitDuplicateObjectExists

A hosting unit object with the same name already exists. Hosting unit names must be unique.

HostingUnitStorageDuplicateObjectExists

A hosting unit storage object with the same storage path already exists for this hosting unit. There can be only one object for each combination of hosting unit and storage path.

HypervisorNotContactable

The hypervisor could not be contacted at the supplied address, which is either invalid or unreachable.

HypervisorAddressInvalidFormat

The address is not in a valid form for the specified hypervisor type.

ConnectionAddressInvalid

The specified address is invalid and does not belong to the same pool.

HypervisorConnectionDuplicateObjectExists

A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.

HypervisorConnectionAddressDuplicateObjectExists

A hypervisor connection address object with the same address already exists for this hypervisor connection. There can be only one object for each combination of hypervisor connection and address.

HypervisorConnectionForHostingUnitIsVirtual

The specified hypervisor connection for the hosting unit is a virtual non-persistent connection. A persistent connection is required when creating a hosting unit.

InputNameInvalid

The name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/;:#.*?=<>|[]()'"{}.

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ScopeNotFound

One or more of the scopes nominated for the new connection do not exist.

CannotCreateSecurityGroupHere

Security groups can only be created in a virtual private cloud (VPC).

Get-Item and Get-ChildItem

HypervisorConnectionObjectNotFound

The hypervisor connection object specified in the path could not be found.

HostingUnitObjectNotFound

The specified hosting unit object could not be found.

HypervisorInMaintenanceMode

The hypervisor for the specified connection is currently in maintenance mode and cannot be accessed.

FailureToRetrieveConnectionPassword

The password for the HypervisorConnection cannot be retrieved or decrypted.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

Remove-Item

HostingUnitObjectToDeleteDoesNotExist

The specified hosting unit object does not exist.

HypervisorConnectionObjectToDeleteDoesNotExist

The specified hypervisor connection object does not exist.

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HypervisorConnectionObjectToDeleteIsInUse

The specified hypervisor connection object cannot be deleted because it is being used by one or more hosting units.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

ObjectCannotBeRemoved

The specified object cannot be removed.

Rename-Item

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HostingUnitDuplicateNameExists

A hosting unit object with the same name already exists. Hosting unit names must be unique.

HypervisorConnectionDuplicateNameExists

A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.

InputNameInvalid

The new name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/;:#.*?=<>|[]()'"{}.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

Set-Item

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HostingUnitObjectToUpdateDoesNotExist

The specified hosting unit object does not exist.

HostingUnitNetworkPathInvalid

The new network path specified for the hosting unit is invalid.
Either the network path does not exist or it is not relative to the
root path of the hosting unit.

HypervisorConnectionObjectToUpdateDoesNotExist

The specified hypervisor connection object does not exist.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

DatabaseNotConfigured
The operation could not be completed because the database for the service
is not configured.

DataStoreException
An error occurred in the service while attempting a database operation.
Communication with the database failed for various reasons.

CommunicationError

There was a problem communicating with the remote service.

about_HypHostSnapIn

March 11, 2024

Topic

about_HypHostSnapin

Short Description

The Host Service PowerShell snap-in provides administrative functions for
the Host Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Hyp'.

Long Description

The Host Service PowerShell snap-in enables both local and remote administration of the Host Service. It lets you configure XenDesktop deployments to make use of hypervisors, networks, and storage, and enables browsing of their contents using the Host PowerShell provider (Citrix.HypervisorProvider).

The provider creates a default PSDrive with a drive identifier of 'XDHyp'. This drive provides two root folders:

HostingUnits Folder

Contains a list of all the hosting units that have been defined using the new-Item provider command.

Hosting units define not only a hypervisor connection, but also reference networks and storage. Hosting units are used by the Machine Creation Service to provide the information required to create and manage virtual machines that can be used by other services. A hosting unit references a root path. This is a specific point in the provider connection tree. The hosting unit is constrained to provide only items below this point in the tree. This restricts the locations that the Machine Creation Service can use to create virtual machines and the networks and storage that can be used.

Connections Folder

Contains a list of all the hosting unit connections that are defined using the new-Item provider command.

Change directory to a specific connection and use the dir/Get-ChildItem command to list all of the infrastructure (such as folders, hypervisors, networks, storage, and virtual machines) that is available in the hosting unit to which the connection refers. The paths to these items are used as the input to commands in the Host Service and Machine Creation Service snap-ins.

The contents of the Hosting Unit and Connection folders reflect the content and structure of the hypervisor to which they refer. The item extensions reflect the object type for each item. Not all item types are appropriate for all hypervisor types. Possible item types are:

Virtual Machine (.vm)
Snapshot (.snapshot)
Cluster (.cluster)

Host (.host)
HostGroup (.hostgroup)
DataCenter (.datacenter)
Folder (.folder)
ResourcePool (.resourcepool)
ComputeResource (.computeresource)

When used with a path that refers to the Host provider (with a default drive of 'XDHyp:'), the Host provider extends the standard New-Item, Get-Item, Get-ChildItem, Remove-Item, Rename-Item, and Set-Item commands as described below. For more information about the basic behavior of these commands, see the help for the command. The Move-Item and Copy-Item commands are not supported for the Host provider.

New-Item

The following parameters are available when using New-Item in the Connections directory (or a path that refers to the directory). The Credential parameter is not supported.

-Name

Specifies the name of the connection, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the connection name must be specified as part of the Path parameter.

-HypervisorAddress <String[]>

Specifies an array of addresses that can be used to contact the required hypervisor. All the addresses are considered equivalent, that is, all of the addresses provide access to the same virtual machines, snapshots, network, and storage.

-ConnectionType <ConnectionType>

Specifies the type of hypervisor that the connection is for. Supported hypervisor types are:

XenServer
SCVMM (Microsoft Hyper-V)
vCenter (VMware vSphere/ESX)
Custom

-PluginId <String>

Specifies the class name for the Citrix Managed Machine library that is used to access the hypervisor. You can obtain this list

using the Get-HypHypervisorPlugin command. The PluginId parameter must be specified if the ConnectionType is set to 'Custom'.

-UserName <String>, -Password <String>, -SecurePassword <SecureString>

Specifies the credentials for the connection to the hypervisor. You can specify the password as either Password or SecurePassword, but not both. The UserName parameter, and either Password or SecurePassword, specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

-HypervisorCredential <PSCredential>

Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as UserName and either Password or SecurePassword, so use of one precludes use of the other.

-Persist

Specifies that the connection details are persistent. If this parameter is not included, the connection is held only for the duration of the current runspace and PSDrive combination. Only persistent connection items are visible to administrators in other runspaces and PSDrives. Hosting units cannot be created from connections that are not persistent.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-Scopes <String[]>

Specifies the list of administrative scopes that the new connection will be a part of. The scopes control which administrators are able to work with the connection.

-ZoneUid <Guid>

Specifies the ZoneUid the connection belongs to. The zone determines which connections are local to the controller.

-IncludeUnavailable <Boolean>

Specifies whether or not to show all hypervisor plug-ins regardless of availability.

The following parameters are available when using New-Item in the HostingUnits directory (or a path that refers to the directory).

-Name

Specifies the name of the hosting unit, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the hosting unit name must be provided as part of the Path parameter.

-HypervisorConnectionName <String>

Specifies the name of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

-HypervisorConnectionUid <Guid>

Specifies the unique identifier of the connection that the hosting unit uses. To create a hosting unit, the referenced connection must be persistent and the ConnectionType must not be set to 'Custom'.

-RootPath <String>

Specifies the location in a connection that is used as the starting reference for the hosting unit. The path must point to an item in the connection that is marked as a SymLink. The root of a XenServer connection is a special case that is also considered a SymLink. If this parameter is not included, the current location in the provider is used.

-NetworkPath <String>

Specifies the path in a connection to the network item that is used when the Machine Creation Service creates new virtual machines.

-StoragePath <String>

Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates new virtual machines. After they are set, you can modify storage paths using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands. If the connection is based on cloud infrastructure, storage items are typically not available, in which case this parameter can be omitted.

-PersonalvDiskStoragePath <String>

Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates disks for the virtual machines. After they are set, you can modify storage paths using the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands.

-NoVmTagging

Specifies that new virtual machines are not tagged with metadata from the hypervisor. By default, all virtual machines created by the Machine Creation Service are tagged to show they are created by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>

Specifies the address of the Host Service that the command will communicate with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching

When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available, use this parameter to specify that the virtual machines created for this hosting unit will use local storage caching for their disk images.

-GpuGroup

Specifies a path to a GPU Group in a connection that will be used when Machine Creation Services creates VMs. Only a single GPU Group is supported and the value is immutable.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-JobGroup <Guid>

Specifies the Uid assocuiated with the HypStorage object created with the New-HypStorage command. This HypStorage object specifies any additional storage details required for the new connection.

The following parameters are available when using New-Item relative to a connection or hosting unit.

-ItemType <String>

Specifies the type of item to create when invoked relative to a connection or hosting unit. Supported ItemType values are:

SecurityGroup (cloud only)

-Description

Specifies a description for the security group.

-VpcId

Specifies a VPC ID for the security group.

Get-Item

Get-ChildItem (alias dir)

Rename-Item

Remove-Item

The following additional parameters are available.

AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection

has not been used, the command attempts to use a local Host Service.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

You can specify the Path parameter by name or by using the unique identifier for the connection or hosting unit enclosed in braces. For example:

-Path XDHyp:{1233-3213ACDF-12323}

The Filter parameter is not supported. Virtual machines created by the Machine Creation Service are returned only if the -Force parameter is used or if virtual machine tagging was disabled on the hosting unit with the NoVmTagging parameter when the VMs were created.

Set-Item

The following parameters are available when using Set-Item in the Connections directory.

-UserName <String>, -Password <String>, -SecurePassword <SecureString>

Specifies the credentials for the connection to the hypervisor. The password can be given as either Password or SecurePassword, but not both. The UserName parameter and either Password or SecurePassword specify the same information as the HypervisorCredential parameter, so use of one precludes use of the other.

-HypervisorCredential <PSCredential>

Specifies credentials for the connection to the hypervisor. The HypervisorCredential parameter specifies the same information as the UserName parameter and either Password or SecurePassword, so use of one precludes use of the other.

-HypervisorAddress <String[]>

Specifies the addresses of the hypervisor that this connection represents. This replaces all existing addresses.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

-MaintenanceMode <Boolean>

Places the connection into maintenance mode which disables all communication between XenDesktop and the Hypervisor. Use this mode when making changes to the hypervisor; for instance, if the password for the access account needs to be changed.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command tries to use a local Host Service.

The Credential parameter is not supported. Addresses can be modified with the Add-HypHypervisorConnectionAddress and Remove-HypHypervisorConnectionAddress commands. Metadata can be modified with the Add-HypMetadata and Remove-HypMetadata commands.

The following parameters are available when using New-Item in the HostingUnits directory.

-Name

Specifies the name of the hosting unit, which must not contain any of the following characters: \/;:#.*?=<>|[]()'"{}. The Name parameter is optional but, if not included, the hosting unit name must be provided as part of the Path parameter.

-NetworkPath <String>

Specifies the path in a connection to the network item that is used when the Machine Creation Service creates new virtual machines.

-NoVmTagging

Specifies that new virtual machines are not tagged with metadata from the hypervisor. By default, all virtual machines created by the Machine Creation Service are tagged to show they are created

by XenDesktop. These tags are used by the provider to restrict the list of virtual machines displayed when viewing the content of the Connections or HostingUnit paths. If this parameter is not included, all virtual machines are displayed at all times.

-AdminAddress <String>

Specifies the address of the Host Service that the command communicates with. After it is set, this address is used for all commands in the Host Service PowerShell snap-in. If this parameter is not included or set by another command, or if Set-HypAdminConnection has not been used, the command attempts to use a local Host Service.

-UseLocalStorageCaching <Boolean>

When Get-HypServiceAddedCapability indicates that the LocalStorageCaching feature is available, use this parameter to specify that the virtual machines created for this hosting unit will use local storage caching for their disk images.

-LoggingId <Guid>

Specifies the identifier of the high-level operation that this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Storage can be modified with the Add-HypHostingUnitStorage and Remove-HypHostingUnitStorage commands. Metadata can be modified with the Add-HypMetadata and Remove-HypMetadata commands.

The item types returned when these commands are used in the Host Service provider are defined in the Object Definitions section below.

Examples

To Create a Persistent Hypervisor Connection

new-item -path “xdhyp:\Connections”-Name MyConn -ConnectionType

XenServer -HypervisorAddress “http:\\address”-UserName user
-Password password -Persist

PSPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\Connections\MyConn

PSParentPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\Connections

PSChildName: MyConn
PSDrive: XDHyp
PSProvider: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor

PSIsContainer: True
HypervisorConnectionUid: 04e6daa2-5cbd-4491-b70c-6daf733ee82a
HypervisorConnectionName: MyConn
ConnectionType: XenServer
HypervisorAddress: {http:\\address}
UserName: user
Persistent: True
PluginId: XenFactory
SupportsPvsVMs: True
Revision: 1b0b0d02-bc1b-49d8-b2b0-be6fb7f150ad
MaintenanceMode: False
Metadata: {MaxAbsoluteActiveActions = 100,

MaxAbsoluteNewActionsPerMinute = 100,
MaxPowerActionsPercentageOfDesktops = 20}

To Create a Hosting Unit

new-item -Path “xdhyp:\HostingUnits”

-Name MyHU
-HypervisorConnectionName MyConn
-RootPath XDHYP:\Connections\MyConn
-NetworkPath XDHYP:\Connections\MyConn\Network 0.network
-StoragePath XDHYP:\Connections\MyConn\Local storage on
myXenServer.storage

PSPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\HostingUnits\MyHU

PSParentPath: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor::XDHyp:\HostingUnits

PSChildName: MyHU
PSDrive: XDHyp

PSProvider: Citrix.HostingUnitService.Admin.V1.0\

Citrix.Hypervisor

PSIsContainer: True
HostingUnitUid: df91f886-1141-4280-bd59-2ee260a4df79
HostingUnitName: MyHU
HypervisorConnection: MyConn
RootPath: /
RootId:
NetworkPath: /Network 0.network
NetworkId: ab47080b-ca15-771a-c8dc-6ad9650158f1
Storage: {/Local storage on myXenServer.storage}
VMTaggingEnabled: True
Metadata: {}

Relative paths can be used for all parameters.

Object Definitions

Citrix.XDInterServiceTypes.HypervisorConnection

The hypervisor connection object is returned when a connection item is located. This item has the following parameters.

HypervisorConnectionUid <Guid>

Specifies the unique identifier for the connection item.

HypervisorConnectionName <String>

Specifies the name of the connection item.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>

Specifies the type of hypervisor that the connection is for.
Supported hypervisor types are:

XenServer
SCVMM (Microsoft Hyper-V)
vCenter (VMware vSphere/ESX)
Custom

HypervisorAddress <String[]>

Specifies the addresses that can be used to contact the required hypervisor.

Username <String>

Specifies the administrator user name for the connection to the hypervisor (from the user name and password given as administrator credentials when

setting up this connection)

Persistent <Boolean>

Specifies whether or not the connection is persistent.

PluginId <String>

Specifies the Citrix Managed Machine class identifier for the hypervisor.

SupportsPvsVM <Boolean>

Specifies whether or not the connection can be used as part of a hosting unit. If this parameter is set to 'True', a hosting unit referencing this connection can be created.

Revision <Guid>

A unique identifier that is changed every time any properties of the connection are changed.

MaintenanceMode <Boolean>

Specifies whether or not the connection is currently in maintenance mode.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>

The collection of metadata associated with the connection.

Citrix.XDInterServiceTypes.HostingUnit

The hosting unit object is returned when a hosting unit item is located. This item has the following parameters.

HostingUnitName <String>

Specifies the name of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HypervisorConnection <Citrix.XDInterServiceTypes.HypervisorConnection>

Specifies the hypervisor connection item that the hosting unit references.

NetworkId <String>

Specifies the unique identifier for the network in the hypervisor context that the hosting unit references.

NetworkPath <String>

Specifies the path to the network in the Host Service provider.

RootId <String>

Specifies the unique identifier for the root connection item in the hypervisor context that the hosting unit references.

RootPath <String>

Specifies the path to the root of the hosting unit from within the Host Service provider.

Storage <Citrix.XDInterServiceTypes.Storage[]>

Specifies the collection of storage objects that are defined for use as part of the hosting unit. This object contains the following parameters.

StorageID <String>

Specifies the identifier for the storage in the hypervisor context.

StoragePath <String>

Specifies the path to the storage in the Host Service provider.

VMTaggingEnabled <Boolean>

Specifies whether or not virtual machine metadata is used to tag virtual machines created by XenDesktop.

Metadata <Citrix.XDInterServiceTypes.Metadata[]>

Specifies the collection of metadata associated with the hosting unit.

Citrix.HostingUnitService.Sdk.HypervisorObject

The hypervisor object is returned when any item is located from within a Connection or HostingUnit folder. This item has the following parameters.

AdditionalData <Dictionary<String,String>

Stores extra untyped data for the item. This dictionary contains
different information for each item type.

For Storage Items

Key = StorageType
Values = Shared or Local

For VM Items

Key = PowerState
Values = PoweredOn,

PoweredOff,
Suspended,
TransitioningToOn,
TransitioningToOff,
Suspending,
Resuming,
Unknown,
Error

Key = TemplateIsWindowsTemplate
Values = True or False

For Snapshot Items

Key = TemplateIsWindowsTemplate
Values = True or False

For Azure Managed Disk Items

Key = TemplateIsWindowsTemplate
Values = True or False

For Azure SIG Image Definition Items

Key = TemplateIsWindowsTemplate
Values = True or False

Name <String>

Specifies the name of the item.

FullName <String>

Specifies the name with the appropriate file extension.

ObjectPath <String>

Specifies the relative path to the item from the root of the
connection of which the item is part.

FullPath <String>

Specifies the absolute path to the item in the Citrix.Hypervisor provider.

Id <String>

Specifies the identity of the item in the hypervisor context.

IsContainer <Boolean>

Specifies whether or not the item can contain other items.

IsSymLink <Boolean>

Specifies whether or not the item can be used as a RootPath of a hosting unit.

ObjectType <Citrix.HostingUnitService.Sdk.NodeType>

Specifies the item type.

Error Codes

The provider commands can return the following error codes.

New-Item

ConnectionNameOrUidInvalid

The name or unique identifier of the hypervisor connection specified for the hosting unit is invalid.

HostingUnitRootPathInvalid

The root path specified for the hosting unit is invalid. The root path must be a valid item in the hypervisor tree and a SymLink.

HostingUnitNetworkPathInvalid

The network path specified for the hosting unit is invalid. The network path must be a valid item in the hypervisor tree and relative to the root path.

HostingUnitStoragePathInvalid

The storage path specified for the hosting unit is invalid. The storage path must be a valid item in the hypervisor tree and relative to the root path.

HostingUnitDuplicateObjectExists

A hosting unit object with the same name already exists. Hosting unit names must be unique.

HostingUnitStorageDuplicateObjectExists

A hosting unit storage object with the same storage path already exists for this hosting unit. There can be only one object for each combination of hosting unit and storage path.

HypervisorNotContactable

The hypervisor could not be contacted at the supplied address, which is either invalid or unreachable.

HypervisorAddressInvalidFormat

The address is not in a valid form for the specified hypervisor type.

ConnectionAddressInvalid

The specified address is invalid and does not belong to the same pool.

HypervisorConnectionDuplicateObjectExists

A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.

HypervisorConnectionAddressDuplicateObjectExists

A hypervisor connection address object with the same address already exists for this hypervisor connection. There can be only one object for each combination of hypervisor connection and address.

HypervisorConnectionForHostingUnitIsVirtual

The specified hypervisor connection for the hosting unit is a virtual non-persistent connection. A persistent connection is required when creating a hosting unit.

InputNameInvalid

The name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/;:#.*?=<>|[]()'"{}.

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ScopeNotFound

One or more of the scopes nominated for the new connection do not exist.

CannotCreateSecurityGroupHere

Security groups can only be created in a virtual private cloud (VPC).

Get-Item and Get-ChildItem

HypervisorConnectionObjectNotFound

The hypervisor connection object specified in the path could not be found.

HostingUnitObjectNotFound

The specified hosting unit object could not be found.

HypervisorInMaintenanceMode

The hypervisor for the specified connection is currently in maintenance mode and cannot be accessed.

FailureToRetrieveConnectionPassword

The password for the HypervisorConnection cannot be retrieved or decrypted.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

Remove-Item

HostingUnitObjectToDeleteDoesNotExist

The specified hosting unit object does not exist.

HypervisorConnectionObjectToDeleteDoesNotExist

The specified hypervisor connection object does not exist.

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HypervisorConnectionObjectToDeleteIsInUse

The specified hypervisor connection object cannot be deleted because it is being used by one or more hosting units.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

ObjectCannotBeRemoved

The specified object cannot be removed.

Rename-Item

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HostingUnitDuplicateNameExists

A hosting unit object with the same name already exists. Hosting unit names must be unique.

HypervisorConnectionDuplicateNameExists

A hypervisor connection object with the same name already exists. Hypervisor connection names must be unique.

InputNameInvalid

The new name specified for the hosting unit or hypervisor connection is invalid because it contains one or more of the following characters: \/;:#.*?=<>|[]()'"{}.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

CommunicationError

There was a problem communicating with the remote service.

Set-Item

InputPathInvalid

The specified path is invalid because it is not in one of the following formats:

XDHyp:\Connections\<Name>
XDHyp:\Connections{Guid}
XDHyp:\HostingUnits\<Name>
XDHyp:\HostingUnits{Guid}

HostingUnitObjectToUpdateDoesNotExist

The specified hosting unit object does not exist.

HostingUnitNetworkPathInvalid

The new network path specified for the hosting unit is invalid. Either the network path does not exist or it is not relative to the root path of the hosting unit.

HypervisorConnectionObjectToUpdateDoesNotExist

The specified hypervisor connection object does not exist.

ExceptionThrown

An unexpected error occurred.

DatabaseError

There was a problem communicating with the database.

DatabaseNotConfigured
The operation could not be completed because the database for the service is not configured.

DataStoreException
An error occurred in the service while attempting a database operation. Communication with the database failed for various reasons.

CommunicationError

There was a problem communicating with the remote service.

about_HypStorage

March 11, 2024

Topic

about_HypStorage

Short Description

The HypStorage object defines parameters for a hosting unit. The parameters specify a storage tier definition made up of the storage type and one or more storage locations. When created it is assigned a job group uid with the -JobGroup parameter.

The HypStorage can then be used with a subsequent New-Item operation via the -JobGroup reference.

Long Description

A HypStorage object defines a set of parameters for a hosting unit. The parameters specify a storage tier definition and the HypStorage can then be used with a subsequent New-Item operation via the -JobGroup reference.

-JobGroup <Uid>

Specifies the unique id of the new job group

-StoragePath <String>

Specifies one or more paths in a connection to storage items that are used when the Machine Creation Service creates new virtual machines.

-StorageType <storage type>

Specifies the type of the new storage tier. Currently the only storage type supported is TemporaryStorage.

Once added to the HostingUnit with the New-Item cmdlet, the additional storage can subsequently be modified or removed with the Add-HypHostingUnitStorage, Set-HypHostingUnitStorage and Remove-HypHostingUnitStorage cmdlets.

Examples

To create a HypStorage and use it when defining a new Hosting Unit

Get A Unique Id For The Job Group

$job = [Guid]::NewGuid()

Define The Storage Tier

New-HypStorage -JobGroup $job -StorageType TemporaryStorage -StoragePath @('XDHyp:\Connections\MyConn
Storage.storage')

Use It In The New-Item Call

New-Item -HypervisorConnectionName 'MyConnection'-JobGroup $job -NetworkPath @('XD-Hyp:\Connections\MyConnection\Network 1.network') <additional New-Item parameters…>

Related Links

about_HypHostSnapIn
Add-HypHostingUnitStorage
Remove-HypHostingUnitStorage

Set-HypHostingUnitStorage
New-Item

about_Hyp_CustomProperties

March 11, 2024

Topic

XenServer SDK - Custom Properties

Short Description

Overview of custom properties settings for a HypervisorConnection object.

Long Description

Custom properties are properties of the HypervisorConnection object, which are specific to the object's hosting hypervisor infrastructure. Custom properties can be used to control or tune behaviors of the hosting connection.

Custom Properties For Azure Resource Manager

The following custom properties are specific to AzureRM.

- SubscriptionId: (String) The Azure Subscription ID.
- TenantId: (String) The Azure Tenant ID.
- AuthenticationAuthority: (String) The Microsoft user authentication endpoint.
  e.g. "https://login.microsoftonline.com/"
- ManagementEndpoint: (String) The Azure Resource Manager endpoint used to establish communications with Azure APIs.
  e.g. "https://management.azure.com/"

- StorageSuffix: (String) The DNS suffix of the Azure Storage endpoint URL. There are 4 default suffix strings, each one belonging to a different Azure environment:
  Azure Global: "core.windows.net"
  Azure China: "core.chinacloudapi.cn"
  Azure US Government: "core.usgovcloudapi.net"
  Azure Germany: "core.cloudapi.de"
- ResourceManagementOperationPollingInterval: (Int) How long Azure waits before polling the status of all running operations managed by Azure's ResourceManagementClient class. Default is 3 seconds in order to poll often without throttling Azure by polling more often than necessary.
- MaximumConcurrentProvisioningOperations: (Int) The number of threads to use within the hypervisor manager, which limits number of simultaneous MCS Operations that are permitted in the provisioning workflow. Default is 500 threads for Azure, where each thread manages exactly one MCS Operation. Increasing the number of threads is not recommended.
- PowerStateCacheExpirationInMinutes: (Int) The number of minutes until the memory cache, where the power state is stored, is expired. Once expired, one Azure API call will be made to get the latest power status for all Azure related catalogs in the site. This effectively refreshes the memory cache. Default is 5 mins.
- PowerStateCacheDegreeOfParallelism: (Int) The number of threads that should be used to fetch missing power states, individually, in parallel, in the event where some machines may be missing from an Azure Resource Graph (ARG) response. In such cases, power states for machines are fetched individually, either through another ARG query or a Virtual Machine 'Get'call. Default number of threads is 10. The number of threads provided as an argument must be between 1 and 512.
- PowerStateCacheUseCachedValuesOnlyAfterInSeconds: (Int) The number of seconds that can elapse before individual 'Get'state calls to Azure are no longer queried and instead machine power states are returned from the memory cached.

  Querying machine power states follows a three step process:

  1. A single Azure Resource Graph (ARG) call is used to query all machine

  power states.

  1. Machine power states that were not picked up by ARG are subsequently

individually queried by Azure 'Get'state calls.

1. Machine power states not successfully queried by Azure 'Get'state

calls within this custom property's timeframe will be returned from the memory cache.

If a machine's power state is not found in the memory cache, then it will be reported as Unknown. Default is 180 seconds. This property does not check if any query has timed out.

Custom Properties For Aws

The following custom properties are specific to the HypervisorConnection object using the AWS plugin.

- MaximumConcurrentProvisioningOperations: (Int) The number of threads to use within the hypervisor manager, which limits number of simultaneous MCS Operations that are permitted in the provisioning workflow. Default is 100 threads for AWS, where each thread manages exactly one MCS Operation. Increasing the number of threads is not recommended.

Custom Properties For Gcp

The following custom properties are specific to the HypervisorConnection object using the GCP plugin.

- MaximumConcurrentProvisioningOperations: (Int) The number of threads to use within the hypervisor manager, which limits number of simultaneous MCS Operations that are permitted in the provisioning workflow. Default is 500 threads for GCP, where each thread manages exactly one MCS Operation. Increasing the number of threads is not recommended.

See Also

about_Hyp_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying

the name with an ordered list of elements or their numeric values,
or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and
supports combination of conditions with -and and -or, and grouping
with braces. For example:

Get-<Noun> -Filter 'Name -like ''High*''-or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use
script blocks in most cases. This can be easier to read as it reduces
quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double
quotes, and omit the quotes altogether for many strings. The order of
parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company ''citrix''-Product '[X]EN*'
Get-<Noun> -Product ''Xen*''-Company ''CITRIX''
Get-<Noun> -Filter { Company -eq 'Citrix'-and Product -like 'Xen*'}

See about_Quoting_Rules and about_Wildcards for details about PowerShell
handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape
the wildcards using the normal PowerShell escape mechanisms (see
about_Escape_Characters), or switch to a filter expression and the -eq
operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
```



```
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}

By their nature, floating point values, DateTime values, and TimeSpan

values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```


Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid


```
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-HypHostingUnitMetadata

March 11, 2024

Adds metadata on the given HostingUnit.

Syntax

```
Add-HypHostingUnitMetadata
   [-HostingUnitUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-HypHostingUnitMetadata
   [-HostingUnitUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-HypHostingUnitMetadata
   [-HostingUnitName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-HypHostingUnitMetadata
   [-HostingUnitName] <String>
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHostingUnitMetadata
    [-InputObject] <HostingUnit[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHostingUnitMetadata
    [-InputObject] <HostingUnit[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given HostingUnit objects. This cmdlet will not overwrite existing metadata on an object - use the Set-HypHostingUnitMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the HostingUnit with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Add-HypHostingUnitMetadata -HostingUnitUid 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-HostingUnitUid

Id of the HostingUnit

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HostingUnitName

Name of the HostingUnit

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HostingUnit[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the HostingUnit specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1"= "val1"; "name2"= "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

Type:	PSObject
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.Metadata

Add-HypHostingUnitMetadata returns an array of objects containing the new definition of the metadata.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypHostingUnitMetadata
- Remove-HypHostingUnitMetadata

Add-HypHostingUnitNetwork

March 11, 2024

Makes additional hypervisor networks available for use in a HostingUnit.

Syntax

```
Add-HypHostingUnitNetwork
   [-NetworkPath] <String>
   [-LiteralPath] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to extend the set of hypervisor networks that are made available through the HostingUnit to the Citrix Machine Creation Service. When new machines are created, their virtual NICs can be associated only with networks that are in this set. This command cannot be used if the connection for the hosting unit is in maintenance mode.

Examples

EXAMPLE 1

The command adds a new network called “newNetwork.network”to the hosting unit called “MyHostingUnit”.

```
Add-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\
   MyHostingUnit -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\
   newNetwork.network'

          HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
          HostingUnitName      : MyHostingUnit
          HypervisorConnection : MyConnection
          RootPath             : /
          RootId               :
          NetworkPath          : /Network 0.network
          NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
          Storage              : {
 /Local storage.storage }

          PersonalvDiskStorage : {
  }

          VMTaggingEnabled     : True
          Metadata             : {
  }

          PermittedNetworks    : {
 /Network 0.network, /newNetwork.network }
```

EXAMPLE 2

The command adds a new network called “newNetwork.network”to the current directory. The dot (.) represents the current location (not its contents).

```
XDHyp:\HostingUnits\MyHostingUnit>Add-HypHostingUnitNetwork -
   LiteralPath . -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\
   newNetwork.network'

          HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
          HostingUnitName      : MyHostingUnit
          HypervisorConnection : MyConnection
          RootPath             : /
          RootId               :
          NetworkPath          : /Network 0.network
          NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
          Storage              : {
 /Local storage.storage }

```



```
13            PersonalvDiskStorage : {
14    }
15
16            VMTaggingEnabled     : True
17            Metadata             : {
18    }
19
20            PermittedNetworks    : {
21   /Network 0.network, /newNetwork.network }
```

EXAMPLE 3

The command adds all of the networks that are available in the hosting unit to the specified hosting unit.

```
1  XDHyp:\HostingUnits\MyHostingUnit>dir *.network | Add-
       HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\MyHostingUnit
```

EXAMPLE 4

The command adds a new network location called “newNetwork.network”to the hosting unit called “MyHostingUnit”.

```
1  Add-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\
       MyHostingUnit -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\
       newNetwork.network'
2
3  HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
4  HostingUnitName      : MyHostingUnit
5  HypervisorConnection : MyConnection
6  RootPath             : /
7  RootId               :
8  NetworkPath          : /Network 0.network
9  NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
10 Storage              : {
11  /Local storage.storage }
12
13 PersonalvDiskStorage : {
14   }
15
16 VMTaggingEnabled     : True
17 Metadata             : {
18   }
19
20 PermittedNetworks    : {
21  /Network 0.network, /newNetwork.network }
```


Parameters

-LiteralPath

Specifies the path to the hosting unit to which the network will be added. The path must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkPath

Specifies the path to the network that will be added. The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>\MyNetwork.network

or <drive>:\Connections{<Connection Uid>}\MyNetwork.network

or <drive>:\HostingUnits\<HostingUnitName>\MyNetwork.network

or <drive>:\HostingUnits{<hostingUnit Uid>}\MyNetwork.network

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Add-HypHostingUnitNetwork (NetworkPath parameter).

Outputs

Citrix.Host.Sdk.HostingUnit

Add-HypHostingUnitNetwork returns an object containing the new definition of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.Host.Sdk.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.Host.Sdk.Storage[]>

The list of storage items that the hosting unit can use.

PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether or not the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Service.

NetworkId <string>

The hypervisor's internal identifier that represents the default network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the default network specified for the hosting unit.

Metadata <Citrix.Host.Sdk.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

PermittedNetworks <Citrix.Host.Sdk.Network[]>

A full list of the hypervisor networks that are exposed for use in the hosting unit.

Notes

The network path must be valid for the hosting unit. The rules that are applied are as follows:

XenServer (HypervisorConnection Type = XenServer)

NA

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)

The network path must be directly contained in the root path item of the hosting unit.

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)

NA

In the case of failure, the following errors can result.

Error Codes

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitNetworkPathInvalid

The specified path is invalid.

HostingUnitNetworkPathInvalid

The network path cannot be found or is invalid. See notes above about validity.

HostingUnitNetworkDuplicateObjectExists

The specified network path is already part of the hosting unit.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item
- Add-HypMetadata
- remove-HypHostingUnitNetwork

Add-HypHostingUnitStorage

March 11, 2024

Adds storage locations to a hosting unit.

Syntax

```
Add-HypHostingUnitStorage
   [-StoragePath] <String>
   [-StorageType <StorageType>]
   [-LiteralPath] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to add storage locations for storing the hard disks required by the virtual machines created by the Citrix Machine Creation Service. You cannot use this command if the connection for the hosting unit is in maintenance mode.

Examples

EXAMPLE 1

The command adds a new storage location called "newStorage.storage"to the hosting unit called "MyHostingUnit".

```
Add-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\
   MyHostingUnit -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\
   newStorage.storage'

          HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
          HostingUnitName      : MyHostingUnit
          HypervisorConnection : MyConnection
          RootPath             : /
          RootId               :
          NetworkPath          : /Network 0.network
          NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
          Storage              : {
 /Local storage.storage, /newStorage.storage }

          PersonalvDiskStorage : {
 /newStorage.storage }

          VMTaggingEnabled     : True
          AdditionalStorage    : {
 TemporaryStorage }

          Metadata             : {
  }
```

EXAMPLE 2

The command adds a new storage location called "newStorage.storage"to the current directory. The dot (.) represents the current location (not its contents).

```
XDHyp:\HostingUnits\MyHostingUnit>Add-HypHostingUnitStorage -
   LiteralPath . -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\
   newStorage.storage' -StorageType OSStorage

          HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
          HostingUnitName      : MyHostingUnit
          HypervisorConnection : MyConnection
          RootPath             : /
          RootId               :
          NetworkPath          : /Network 0.network
          NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
          Storage              : {
 /Local storage.storage, /newStorage.storage }

```



```
13            PersonalvDiskStorage : {
14   /Local storage.storage }
15
16            VMTaggingEnabled     : True
17            AdditionalStorage    : {
18   TemporaryStorage }
19
20            Metadata             : {
21    }
```

EXAMPLE 3

The command adds all of the storage that is available in the hosting unit to the specified hosting unit.

```
1  XDHyp:\HostingUnits\MyHostingUnit>dir *.storage | Add-
       HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\MyHostingUnit
```

EXAMPLE 4

The command adds a new storage location called “newStorage.storage”to the hosting unit called “MyHostingUnit”.

```
1  Add-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\
       MyHostingUnit -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\
       newStorage.storage' -StorageType PersonalvDiskStorage
2
3  HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
4  HostingUnitName      : MyHostingUnit
5  HypervisorConnection : MyConnection
6  RootPath             : /
7  RootId               :
8  NetworkPath          : /Network 0.network
9  NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
10 Storage              : {
11  /Local storage.storage }
12
13 PersonalvDiskStorage : {
14  /Local storage.storage, /newStorage.storage }
15
16 VMTaggingEnabled     : True
17 AdditionalStorage    : {
18  TemporaryStorage }
19
20 Metadata             : {
21   }
```


Parameters

-LiteralPath

Specifies the path to the hosting unit to which storage will be added. The path must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StoragePath

Specifies the path to the storage that will be added. The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>\MyStorage.storage

or <drive>:\Connections{<Connection Uid>}\MyStorage.storage

or <drive>:\HostingUnits\<HostingUnitName>\MyStorage.storage

or <drive>:\HostingUnits{<hostingUnit Uid>}\MyStorage.storage

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-StorageType

Specifies the type of storage in StoragePath. Supported storage types are:

OSStorage

PersonalvDiskStorage

TemporaryStorage

Type:	StorageType
Position:	Named
Default value:	OSStorage
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Add-HypHostingUnitStorage (StoragePath parameter).

Outputs

Citrix.Host.Sdk.HostingUnit

Add-HypHostingUnitStorage returns an object containing the new definition of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.Host.Sdk.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.Host.Sdk.Storage[]>

The list of storage items that the hosting unit can use.

PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether or not the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Service.

NetworkId <string>

The hypervisor's internal identifier that represents the network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the network specified for the hosting unit.

AdditionalStorage

The list of additional storage items that the hosting unit can use.

Metadata <Citrix.Host.Sdk.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

Notes

The storage path must be valid for the hosting unit. The rules that are applied are as follows:

XenServer (HypervisorConnection Type = XenServer)

NA

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)

The storage path must be directly contained in the root path item of the hosting unit.

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)

Only one storage entry for these connection types is valid, and it must reference an SMB share. Additionally, if a Hyper-V failover cluster is used the SMB share must be the top-level mount point of the cluster shared volume on one of the servers in the cluster (i.e. C:\ClusterStorage).

In the case of failure, the following errors can result.

Error Codes

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitStoragePathInvalid

The specified path is invalid.

HostingUnitStoragePathInvalid

The storage path cannot be found or is invalid. See notes above about validity.

HostingUnitStorageDuplicateObjectExists

The specified storage path is already part of the hosting unit.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item
- Add-HypMetadata
- Remove-HypHostingUnitStorage
- Set-HypHostingUnitStorage

Add-HypHypervisorConnectionAddress

March 11, 2024

Add a connection address to a hypervisor connection.

Syntax

```
Add-HypHypervisorConnectionAddress
   [-LiteralPath] <String>
   [-HypervisorAddress] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to add addresses by which a hypervisor can be contacted. All addresses added to a single hypervisor connection are assumed to be equivalent (i.e. they all result in the ability to communicate with the same hypervisors). The hypervisor that the addresses reference is stored at the point of creation of the hypervisor connection. Once this is done, to be valid, all addresses must resolve to this hypervisor.

The addresses required are;

XenServer - The address of the XenServer machines (must all reference the same XenServer pool).

VMWare vSphere/ESX - The address of a vCenter Server.

Microsoft SCVMM/Hyper-V - the address of an SCVMM server.

Examples

EXAMPLE 1

Add the address ‘http:\\myserver.com’to the hypervisor connection called “MyConnection”.

```
Add-HypHypervisorConnectionAddress -LiteralPath XDHyp:\Connections\
   MyConnection -HypervisorAddress http:\\myserver.com

PSPath                   : Citrix.HostingUnitService.Admin.V1.0\Citrix.
   Hypervisor::XDHyp:\Connections\MyConnection
PSParentPath             : Citrix.HostingUnitService.Admin.V1.0\Citrix.
   Hypervisor::XDHyp:\Connections
PSChildName              : MyConnection
PSDrive                  : XDHyp
PSProvider               : Citrix.HostingUnitService.Admin.V1.0\Citrix.
   Hypervisor
PSIsContainer            : True
HypervisorConnectionUid  : 85581f42-c5da-4976-970c-ebc3448ea1e3
HypervisorConnectionName : MyConnection
ConnectionType           : XenServer
HypervisorAddress        : {
 http:\\myserver2.com,http:\\myserver.com }
```



```
14
15 UserName                : root
16 Persistent              : False
17 PluginId                : XenFactory
18 SupportsPvsVMs          : True
19 Revision                : 4c95c857-c54d-4f92-abef-0cce32c02502
20 Metadata                :
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hypervisor connection item to which to add the address. The path specified must be in one of the following formats;

<drive>:\Connections\<HypervisorConnectionName>

or <drive>:\Connections{HypervisorConnection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorAddress

Specifies the address to be used. The address will be validated and the hypervisor must be contactable at the address supplied.

XenServer (ConnectionType = XenServer)

The address being added must reference the same XenServer pool referenced by any existing addresses for the same connection.

Type:	String
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.HypervisorConnection

Add-HypHypervisorConnectionAddress returns an object containing the new definition of the hypervisor connection.

HypervisorConnectionUid <Guid>

Specifies the unique identifier for the hypervisor connection.

HypervisorConnectionName <string>

Specifies the name of the hypervisor connection.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>

Specifies the connection type of the connection.

XenServer - A XenServer hypervisor

SCVMM - A Microsoft SCVMM/Hyper-V hypervisor

vCenter - A VMWare vSphere/ESX hypervisor

Custom - A 3rd party hypervisor

HypervisorAddress <string[]>

A list of addresses that can be used to communicate with the hypervisor.

UserName <string>

The user name that is used when connecting to the hypervisor.

Persistent <Boolean>

Indicates whether the connection is stored in the database or is a temporary connection only with the same lifetime as the current Powershell session.

PlugInId <string>

The Citrix identifier for the Citrix Machine Management plug-in.

Revision <Guid>

Identifier for the current version of the hypervisor connection. This value changes every time a property of the hypervisor connection is updated.

SupportsPvsVMs <Boolean>

Indicates whether or not the connection can be used as part of a hosting unit and therefore used by the Citrix XenDesktop Machine Creation Service to create virtual machines. Only the built-in supported hypervisor connection types can be used for this (i.e. XenServer, SCVMM and vCenter).

Metadata <Citrix.Host.Sdk.Metadata[]>

A list of key value pairs that can store additional information relating to the hosting unit.

Notes

The address format must be valid for the hypervisor connection. The rules that are applied are as below:

XenServer (HypervisorConnection Type = XenServer)

http:\\<IP Address>

or http:\\<server Name>

or https:\\<server Name>

Note: To use multiple addresses to the same XenServer pool to provide failover functionality, all XenServers in the pool must have a shared block storage device. If the use of https connections and failover is required, the certificates on the servers must be trusted by all of the controllers (typically this means having a root certificate installed).

Note: If XenServers are moved to new XenServer pools after being added to a hypervisor connection, unpredictable results can occur. Once a XenServer is assigned to a hypervisor connection, it must not be moved to a new XenServer pool.

In the case of failure the following errors can result.

Error Codes

InputConnectionsPathInvalid

The path provided is not to an item in a sub directory of a hosting unit item.

HypervisorConnectionAddressForeignKeyObjectDoesNotExist

The hypervisor connection to which the address is to be added could not be located.

ConnectionAddressInvalid

The address could not be used to contact a hypervisor or the validation rules have not been met.

HypervisorConnectionAddressDuplicateObjectExists

The address specified is already part of the hypervisor connection.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Remove-HypHypervisorConnectionAddress
- Get-HypXenServerAddress

Add-HypHypervisorConnectionMetadata

March 11, 2024

Adds metadata on the given HypervisorConnection.

Syntax

```
Add-HypHypervisorConnectionMetadata
   [-HypervisorConnectionUid] <Guid>
   -Name <String>
   -Value <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHypervisorConnectionMetadata
    [-HypervisorConnectionUid] <Guid>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHypervisorConnectionMetadata
    [-HypervisorConnectionName] <String>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHypervisorConnectionMetadata
    [-HypervisorConnectionName] <String>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHypervisorConnectionMetadata
    [-InputObject] <HypervisorConnection[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-HypHypervisorConnectionMetadata
    [-InputObject] <HypervisorConnection[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given HypervisorConnection objects. This cmdlet will not overwrite existing metadata on an object - use the Set-HypHypervisorConnectionMetadata

cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the HypervisorConnection with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Add-HypHypervisorConnectionMetadata -HypervisorConnectionUid 4CECC26E
    -48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

Property                                    Value
--------                                    -----
property                                    value
```

Parameters

-HypervisorConnectionUid

Id of the HypervisorConnection

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionName

Name of the HypervisorConnection

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the HypervisorConnection specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.Metadata

Add-HypHypervisorConnectionMetadata returns an array of objects containing the new definition of the metadata.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypHypervisorConnectionMetadata
- Remove-HypHypervisorConnectionMetadata

Add-HypHypervisorConnectionScope

March 11, 2024

Add the specified HypervisorConnection(s) to the given scope(s).

Syntax

```
Add-HypHypervisorConnectionScope
   [-Scope] <String[]>
   -InputObject <HypervisorConnection[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-HypHypervisorConnectionScope
   [-Scope] <String[]>
   -HypervisorConnectionUid <Guid[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-HypHypervisorConnectionScope
   [-Scope] <String[]>
   -HypervisorConnectionName <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-HypHypervisorConnectionScope command is used to associate one or more HypervisorConnection objects with given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the HypervisorConnection objects in different ways:

- HypervisorConnection objects can be piped in or specified by the InputObject parameter
- The HypervisorConnectionUid parameter specifies objects by HypervisorConnectionUid
- The HypervisorConnectionName parameter specifies objects by HypervisorConnectionName (supports wildcards)

To add a HypervisorConnection to a scope you need permission to change the scopes of the HypervisorConnection and permission to add objects to all of the scopes you have specified.

If the HypervisorConnection is already in a scope, that scope will be silently ignored.

Examples

EXAMPLE 1

Adds a single HypervisorConnection to the 'Finance' scope.

```
Add-HypHypervisorConnectionScope Finance -HypervisorConnectionUid 6702
    C5D0-C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 2

Adds a single HypervisorConnection to the multiple scopes.

```
Add-HypHypervisorConnectionScope Finance,Marketing -
    HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 3

Adds all visible HypervisorConnection objects to the 'Finance' scope.

```
Get-HypHypervisorConnection | Add-HypHypervisorConnectionScope Finance
```

EXAMPLE 4

Adds HypervisorConnection objects with a name starting with an 'A' to the 'Finance' scope.

```
Add-HypHypervisorConnectionScope Finance -HypervisorConnectionName A*
```

Parameters

-Scope

Specifies the scopes to add the objects to.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-InputObject

Specifies the HypervisorConnection objects to be added.

Type:	HypervisorConnection[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionUid

Specifies the HypervisorConnection objects to be added by HypervisorConnectionUid.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionName

Specifies the HypervisorConnection objects to be added by HypervisorConnectionName.

Type:	String[]
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Remove-HypHypervisorConnectionScope
- Get-HypScopedObject

Add-HypMetadata

March 11, 2024

Adds metadata to a hypervisor connection or a hosting unit.

Syntax

```
Add-HypMetadata
   [-Property] <String>
   [-Value] <String>
   [-LiteralPath] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to store additional custom data against a hosting unit or hypervisor connection. This data is not used by the Machine Creation Service, and is provided only for consumers of the services to store any data that may be required for their operations. The metadata is returned along with the hypervisor connection or hosting unit that it is assigned to.

Examples

EXAMPLE 1

The command adds the metadata with the property name of “MyProperty”and value of “MyValue”to the hypervisor connection item called “MyConnection”.


```
Add-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection -Property
    MyProperty -Value MyValue

Property                                                        Value
--------                                                        -----
MyProperty                                                      MyValue
```

EXAMPLE 2

The command adds the metadata with the property name of “MyProperty”and value of “MyValue”to all the hypervisor connection items that begin with the string “Citrix”.

```
dir xdhyp\connections\Citrix* | Add-HypMetadata -Property MyProperty -
    Value MyValue

Property                                                        Value
--------                                                        -----
MyProperty                                                      MyValue
MyProperty                                                      MyValue
MyProperty                                                      MyValue
```

Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the hosting unit or hypervisor connection item to which to add the metadata. The path specified must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

or <drive>:\Connections\<Connection Name>

or <drive>:\Connections{<Connection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Property

Specifies the property name of the metadata to be added. The property must be unique for the item specified by the path.

The property cannot contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string the contains a path to Add-HypMetadata (Path parameter).

Outputs

Citrix.Host.Sdk.Metadata

Add-HypMetadata returns an array of objects containing the new definition of the metadata.

- Property <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

In the case of failure, the following errors can result.

Error Codes

InvalidPath

The path provided is not in the required format.

HostingUnitMetadataForeignKeyObjectDoesNotExist

The hosting unit supplied in the path does not exist.

HypervisorConnectionMetadataForeignKeyObjectDoesNotExist

The hypervisor connection supplied in the path does not exist.

HostingUnitMetadataDuplicateObjectExists

Metadata for the specified hosting unit item already exists with the same property name.

HypervisorConnectionMetadataDuplicateObjectExists

Metadata for the specified hypervisor connection item already exists with the same property name.

MetadataContainerUndefined

The specified path does not reference a hosting unit or a hypervisor connection.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Remove-HypMetadata

Get-HypConfigurationDataForItem

March 11, 2024

Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.

Syntax

```
Get-HypConfigurationDataForItem
   [-LiteralPath] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This command provides a mechanism for retrieving extra data about an entry in the hosting unit service provider. The referenced item must be contained within the connections directory in the provider (i.e. XDHyp:\Connections).

This mechanism is used for obtaining data that is not required frequently and/or has a high overhead associated with its retrieval (so as to maintain the responsiveness of the provider). For this release of the PowerShell snap-in, the only provider items that can be used with this operation are VM items. For a VM, this provides a mechanism to obtain the number of CPUs, the amount of Memory (in MB) and hard disk drive capacity (in GB).

Examples

EXAMPLE 1

This command gets the configuration properties for a VM called ‘MyVm.vm’within a hypervisor connection called ‘MyConnection’.


```
Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\
    MyConnection\MyVm.vm

                                CpuCount
                                    MemoryMB
                                    HardDiskSizeGB
                                --------
                                    --------
                                    --------------
                                        1

                                            1024

                                            24
```

EXAMPLE 2

This command gets the configuration properties for a VM called ‘MyVm.vm’within the current directory. The dot (.) represents the current location (not its contents).

```
XDHyp:\HostingUnits\PS>Get-HypConfigurationDataForItem -LiteralPath .\
    MyVm.vm

                                CpuCount
                                    MemoryMB
                                    HardDiskSizeGB
                                --------
                                    --------
                                    --------------
                                        1

                                            1024

                                            24
```

EXAMPLE 3

This command gets the CPU count for a VM called ‘MyVm.vm’. The CPUCount is just one property of the VM items. To see all properties of an item, type “(Get-HypConfigurationDataForItem <ItemPath>) | Get-Member”.

```
(Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\
    MyConnection\MyVm.vm).CPUCount
```


Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the item for which configuration data is to be retrieved. The path specified must be in one of the following formats;

<drive>:\Connections\<Connection Name>\<Item Path of VM object>

or <drive>:\Connections{<connection Uid>\<Item Path of VM object>}

or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object>

or <drive>:\HostingUnits{<hostingUnit Uid>\<Item Path of VM object>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Get-HypConfigurationDataForItem

Outputs

PSObject

Get-HypConfigurationDataForItem returns a PSObject containing the properties that are appropriate for the item specified by the path.

Properties for VM Item ——————- CPUCount <int>

Specifies the number of CPUs assigned to the VM.

MemoryMB <int>

The amount of memory allocated to the VM.

HardDiskSizeGB <int>

The capacity of the primary hard drive assigned to the VM.

Notes

For this release, this cmdlet provides only configuration data for VM objects in the provider. Using a path to an item that is not a VM results in a error.

In the case of failure the following errors can result.

Error Codes

InputHypervisorItemPathInvalid

The path provided is not to an item in a sub-directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

HypervisorPermissionDenied

The hypervisor login used does not provide authorization to access the data for this item.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-Item

Get-HypConfigurationObjectForItem

March 11, 2024

Retrieves the configuration data for an item in the Host Service provider path. Note: For this release, only VM items are supported for this operation.

Syntax

```
Get-HypConfigurationObjectForItem
   [-LiteralPath] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

This command provides a mechanism for retrieving extra data about an entry in the hosting unit service provider. The referenced item must be contained within the connections directory in the provider (i.e. XDHyp:\Connections).

This mechanism is used for obtaining data that is not required frequently and/or has a high overhead associated with its retrieval (so as to maintain the responsiveness of the provider). For this release of the PowerShell snap-in the only provider items that can be used with this operation are VM items. For a VM this provides a mechanism to obtain the number of CPUs, the amount of Memory (in MB) and hard disk drive capacity (GB).

Examples

EXAMPLE 1

This command gets the configuration properties for a VM called 'MyVm.vm'within a hypervisor connection called 'MyConnection'.

```
Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\
    MyConnection\MyVm.vm

                                CpuCount
                                    MemoryMB
                                    HardDiskSizeGB
                                --------
                                    --------
                                    --------------
                                        1

                                            1024

                                            24
```

EXAMPLE 2

This command gets the configuration properties for a VM called 'MyVm.vm'within the current directory. The dot (.) represents the current location (not its contents).

```
XDHyp:\HostingUnits\PS>Get-HypConfigurationDataForItem -LiteralPath .\
    MyVm.vm

                                CpuCount
                                    MemoryMB
                                    HardDiskSizeGB
```



```
4                                   --------
                                        --------
                                        --------------
5                                           1

                                                1024

                                                24
```

EXAMPLE 3

This command gets the CPU count for a VM called 'MyVm.vm'. The CPUCount is just one property of the VM items. To see all properties of an item, type "(Get-HypConfigurationDataForItem <ItemPath>) | Get-Member".

```
1  (Get-HypConfigurationDataForItem -LiteralPath XDHyp:\Connections\
      MyConnection\MyVm.vm).CPUCount
```

Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the item for which configuration data is to be retrieved. The path specified must be in one of the following formats;

<drive>:\Connections\<Connection Name>\<Item Path of VM object>

or <drive>:\Connections{<connection Uid>\<Item Path of VM object>}

or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object>

or <drive>:\HostingUnits{<hostingUnit Uid>\<Item Path of VM object>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string the contains a path to Get-HypConfigurationDataForItem

Outputs

PSObject

Get-HypConfigurationDataForItem returns a PSObject containing the properties that are appropriate for the item specified by the path.

Properties for VM Item ——————- CPUCount <int>

Specifies the number of CPUs assigned to the VM.

MemoryMB <int>

The amount of memory allocated to the VM.

HardDiskSizeGB <int>

The capacity of the primary hard drive assigned to the VM.

Network Map

The networks that this VM or Snaphot is connected to.

Notes

For this release this cmdlet only provides configuration data for VM objects in the provider. Using a path to an item that is not a VM will result in a error.

In the case of failure the following errors can be produced.

Error Codes

InputHypervisorItemPathInvalid

The path provided is not to an item in a sub directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

DatabaseError

An error occurred in the service whilst attempting a database operation.

DatabaseNotConfigured

The operation could not be completed as the database for the service is not configured.

DataStoreException

An error occurred in the service whilst attempting a database operation - communication to database failed for

for various reasons.

CommunicationError

An error occurred whilst communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

HypervisorPermissionDenied

The hypervisor login used does not provide authorization to access the data for this item.

ExceptionThrown

An unexpected error occurred. To locate more details see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-Item

Get-HypConnectionRegion

March 11, 2024

Enumerates the regions of a hypervisor connection that are based on cloud technology.

Syntax

```
Get-HypConnectionRegion
   [-LiteralPath] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to enumerate the available regions within a public or private cloud, when making hypervisor connections to cloud services. Sometimes, regions need to be selected and applied before the cloud connection can be used in a meaningful way. This cmdlet allows the supported regions to be listed so that one may be selected.

Once a region has been chosen, use the standard Set-Item provider cmdlet to select it. See the examples for further details.

This cmdlet may return no output, in which case the cloud connection can be considered "regionless" (or, implicitly, all within a single region). In such cases, there is no need to select a region, and the hypervisor connection can be used as is.

Examples

EXAMPLE 1

This sequence of commands enumerates the available regions of an Amazon AWS cloud connection, and then selects one of them for use in the connection.

```
Get-HypConnectionRegions -LiteralPath XDHyp:\Connections\AWS

          RegionName           : us-east-1
          Endpoint             : ec2.us-east-1.amazonaws.com

          RegionName           : us-west-1
          Endpoint             : ec2.us-west-1.amazonaws.com

          RegionName           : eu-west-1
```



```
10          Endpoint              : ec2.eu-west-1.amazonaws.com
11
12          (...)
13
14          c:\PS>Set-Item -Path XDHyp:\Connections\AWS -Region "us-east
               -1"
```

Parameters

-LiteralPath

Specifies the path to the hypervisor connection whose regions are being examined. This cmdlet is valid only for hypervisor connections that have the UsesCloudInfrastructure flag set to true. The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>

or <drive>:\Connections{<Connection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Get-HypConnectionRegion (LiteralPath parameter).

Outputs

Citrix.Host.Sdk.HypervisorRegion

Get-HypConnectionRegion returns zero or more instances of the HypervisorConnectionRegion object, each of which contain the following properties:

Name <string>

Specifies the unique name of the region.

Endpoint <string>

Specifies the URL endpoint that is specific to the region, if relevant. This may be an empty string, and is returned only for information purposes.

A full list of the hypervisor networks that are exposed for use in the hosting unit.

Notes

In the case of failure, the following errors can be produced.

Error Codes

ConnectionsPathInvalid

The path provided is not to an item in the Connections subdirectory of the host service provider.

HypervisorConnectionObjectNotFound

The path provided could not be resolved to an existing hypervisor connection. The name or GUID is invalid.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

ConnectionIsNotCloud

The hypervisor connection is not associated with cloud infrastructure, making it invalid to enumerate regions.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-Item

Get-HypDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Host Service.

Syntax

```
Get-HypDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets the database connection string from the currently selected Host Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Host Service instance running on controller "controller1.mydomain.net".

```
1  Get-HypDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Host Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the HostService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypDBConnection
- Get-HypServiceStatus
- Test-HypDBConnection

Get-HypDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Host Service.

Syntax

```
Get-HypDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Host Service database schema, add a new Host service to an existing site, remove a Host service from a site, or create a database server logon for a Host service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Host service instance, otherwise the scripts relate to Host service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Host service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user

- Addition of database user to Host service roles

If ScriptType is Evict, the returned script contains:

- Removal of Host service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Host Service and copies it to a file called "C:\HostSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Host service schema.

```
Get-HypDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\HostSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Host service. This can be used when configuring a mirror server for use.

```
Get-HypDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    HostLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Host service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Host services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Host Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Host Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Host Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Host Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Host services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Host Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypDBConnection
- Test-HypDBConnection

Get-HypDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Host Service.

Syntax

```
Get-HypDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets an SQL script that can be used to update the current Citrix Host Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Host Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Host Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Host Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Host Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-HypServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
1 $update = Get-HypDBVersionChangeScript -DatabaseName MyDb -
     TargetVersion 7.40.0.0
2 $update.Script > update_740.sql
```


Parameters

-DatabaseName

The name of the database containing the Citrix Host Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-HypDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Host Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Host services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-HypServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypInstalledDBVersion
- Get-HypServiceStatus
- Get-HypDBSchema

Get-HypHypervisorPlugin

March 11, 2024

Gets the available hypervisor types.

Syntax

```
Get-HypHypervisorPlugin
    [-ZoneUid <Guid>]
    [-IncludeUnavailable <Boolean>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Use this command to retrieve a list of all the available hypervisor types, and their localized names.

Examples

EXAMPLE 1

Get the available hypervisor management plug-ins.

```
Get-HypHypervisorPlugin | Format-Table -AutoSize

           ConnectionType DisplayName              PluginFactoryName
              UsesCloudInfrastructure CitrixVerified
           -------------- -----------              -----------------
              ----------------------- --------------
                 SCVMM Microsoft virtualization MicrosoftPSFactory
                                       False           True
                VCenter VMware virtualization    VmwareFactory
                                         False           True
              XenServer XenServer                XenFactory
                                          False           True
```

EXAMPLE 2

Get the available hypervisor management plug-ins for the zone with the specified ZoneUid.

```
Get-HypHypervisorPlugin -ZoneUid 2534080e-9226-4af2-a2fc-ec6be14f9552
```

EXAMPLE 3

Get all Citrix supported hypervisor plug-ins for the specified ZoneUid.

```
Get-HypHypervisorPlugin -ZoneUid 2534080e-9226-4af2-a2fc-ec6be14f9552 -
    IncludeUnavailable $true
```

Parameters

-ZoneUid

ID of the Zone

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludeUnavailable

Specifies whether or not to show all hypervisor plug-ins regardless of availability.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

Outputs

Citrix.Host.Sdk.HypervisorPlugin

Get-HypHypervisorPlugin returns a list of objects containing the definition of the hypervisor plug-ins.

ConnectionType <Citrix.XDInterServiceTypes.ConnectionType>

The hypervisor connection type. This can be one of the following:

XenServer - XenServer hypervisor

SCVMM - Microsoft SCVMM/Hyper-V

vCenter - VMWare vSphere/ESX

Custom - a third-party hypervisor

DisplayName <string>

The localized display name (localized using the locale of the Powershell snap-in session)

PluginFactoryName <string>

The name of the hypervisor plug-in factory used to manage the hypervisor connections.

UsesCloudInfrastructure <Boolean>

Specifies whether or not the underlying hypervisor plug-in factory employs cloud-hosted infrastructure.

CitrixVerified <Boolean>

Specifies whether or not the underlying hypervisor plug-in is verified by Citrix.

Notes

To use third-party plug-ins, the plug-in assemblies must be installed into the appropriate location on each controller machine that forms part of the Citrix controller site. Failure to do this can result in unpredictable behavior, especially during service failover conditions.

In the case of failure the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item

Get-HypInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Host Service.

Syntax

```
Get-HypInstalledDBVersion
   [-Upgrade]
   [-Downgrade]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Host Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-HypDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Host Service database schema version number.

```
1 Get-HypInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Host Service database schema for which upgrade scripts are supplied.

```
1 Get-HypInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-HypInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Host

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypDBVersionChangeScript
- Get-HypDBSchema

Get-HypInventoryItem

March 11, 2024

Gets the resource items and child items from specified inventory locations using the resource type and a set of filtering and pagination parameters.

Syntax

```
Get-HypInventoryItem
   -ResourceType <String>
    [-ContinuationToken <String>]
    [-MaxRecords <Int32>]
    [-Skip <Int32>]
    [-ContainsName <String>]
    [-ForwardDirection <Boolean>]
    [-Tags <String>]
    [-Id <String>]
    [-AdditionalDataFilter <String>]
    [-Template <String>]
    [-Warn <Boolean>]
    [-Force <Boolean>]
    [-LiteralPath] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The ‘Get-HypInventoryItem‘ cmdlet gets the resource items and child items from specified inventory locations using the resource type and a set of filtering and pagination parameters.

‘Get-HypInventoryItem‘ requires a Hosting Unit path and the ResourceType parameter. The Resource-Type specifies the resource extension, such as ‘template‘, ‘vm‘, etc.

Optional parameters control how many items to return, where to start the list of items and how to filter items so that only a subset are returned.

Examples

EXAMPLE 1

Gets the first template resource from the ctx-test Hosting Unit

```
PS C:\Users\admin> Get-HypInventoryItem -LiteralPath "XDHyp:\
   HostingUnits\ctx-test" -ResourceType vm


AdditionalData : {
 [ServiceOfferingDescription, T3 Micro Instance], [TenancyType, Shared
    ], [NetworkMappings,
                [[["subnet-074242d7489b7209e","us-east-1a.
                   availabilityzone\\10.0.128.0/17
                (vpc-0fa6e41d72507f877).network"]]]], [SecurityGroups,
                    ["private.securitygroup"]]... }

FullName       : aws-test-non-persistent-vda-win2022 (i-0
   d2abbcbc89876ed2).vm
FullPath       :
Id             : i-0d2abbcbc89876ed2
IsContainer    : False
IsMachine      : True
IsSnapshotable : True
IsSymLink      : False
Name           : aws-test-non-persistent-vda-win2022 (i-0
   d2abbcbc89876ed2)
ObjectPath     : /aws-test-non-persistent-vda-win2022 (i-0
   d2abbcbc89876ed2).vm
ObjectType     : Vm
ObjectTypeName : vm

PS C:\Users\admin> Get-HypInventoryItem -LiteralPath "XDHyp:\
   HostingUnits\ctx-test" -ResourceType template


Description             : Canonical, Ubuntu, 16.04 LTS, amd64 xenial
   image build on 2019-06-28
HasPersistentRootVolume : True
IsWindowsTemplate       : True
Owner                   : 679593333241
AdditionalData          : {
 [BlockDeviceCount, 1], [EnaSupport, True], [HibernationSupported,
    False], [Warning, Unable
                          to find snapshot or instance for AMI;]... }

FullName                : ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-
   amd64-server-20190628-d83d0782-cb94-
                          46d7-8993-f4ce15d1a484-ami-0cfee17793b08a293
                              .4 (ami-021d9d94f93a07a43).template
FullPath                :
Id                      : ami-021d9d94f93a07a43
```



```
IsContainer              : False
IsMachine                : False
IsSnapshotable           : False
IsSymLink                : False
Name                     : ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20190628-d83d0782-cb94-46d7-8993-f4ce15d1a484-ami-0cfee17793b08a293.4 (ami-021d9d94f93a07a43)
ObjectPath               : /ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20190628-d83d0782-cb94-46d7-8993-f4ce15d1a484-ami-0cfee17793b08a293.4 (ami-021d9d94f93a07a43).template
ObjectType               : Template
ObjectTypeName           : template

PS C:\Users\admin> Get-HypInventoryItem -LiteralPath "XDHyp:\HostingUnits\ctx-test" -ResourceType template -ContainName persistent | select Name

Name
----
test-non-persistent-vda-win2022 (ami-0a2d913927e0352f3)

PS C:\Users\admin> Get-HypInventoryItem -LiteralPath "XDHyp:\HostingUnits\ctx-test" -ResourceType vm -MaxRecords 100 | select Name

Name
----
test-non-persistent-vda-win2022 (i-0f33601c7439a81aa)
test-Automation (i-021f404ff81e4b3b3)
test-non-persistent-vda-win2022-2 (i-0821ca8ba8c3d171c)
test-Bastion (i-026182c752bd20329)
test-AD (i-09dcfc3180ac723ea)
test-non-persistent-hibernation-vda-win2022 (i-02747245f4fd643e6)
test-Ubuntu-MP-test (i-0f311930f6509955b)
test-persistent-vda (i-00c55f1107b662c04)
aws-test-non-persistent-vda-win2022 (i-0d2abbcbc89876ed2)

PS C:\Users\admin> Get-HypInventoryItem -LiteralPath "XDHyp:\HostingUnits\ctx-test" -ResourceType vm -MaxRecords 100 -Skip 2 | select Name

Name
----
test-non-persistent-vda-win2022-2 (i-0821ca8ba8c3d171c)
test-Bastion (i-026182c752bd20329)
test-AD (i-09dcfc3180ac723ea)
test-non-persistent-hibernation-vda-win2022 (i-02747245f4fd643e6)
test-Ubuntu-MP-test (i-0f311930f6509955b)
test-persistent-vda (i-00c55f1107b662c04)
aws-test-non-persistent-vda-win2022 (i-0d2abbcbc89876ed2)
```


EXAMPLE 2

This will return first 100 launch template versions in that hosting unit.

```
PS C:\Users\admin> Get-HypInventoryItem -ResourceType "
    launchtemplateversion" -LiteralPath "XDHyp:\HostingUnits\ctx-test" -
    MaxRecords 100
```

EXAMPLE 3

This will return 100 launch template versions after skipping first 100 in that hosting unit.

```
PS C:\Users\admin> Get-HypInventoryItem -ResourceType "
    launchtemplateversion" -LiteralPath "XDHyp:\HostingUnits\ctx-test" -
    MaxRecords 100 -Skip 100
```

EXAMPLE 4

Examples of Additional Data returned for launch template version and different filters and Parameters.

```
PS C:\Users\admin> $items = Get-HypInventoryItem -ResourceType "
    launchtemplateversion" -LiteralPath "XDHyp:\HostingUnits\ctx-test" -
    MaxRecords 200
PS C:\Users\admin> $items[0].AdditionalData
Key                         Value
---                         -----
ServiceOfferingDescription T3 Large Instance
TenancyType                 Shared
HibernationSupported        False
NetworkMappings             [[["subnet-074242d7489b7209e","us-east-1a.
    availabilityzone\\10.0.128.0/17 (vpc-0fa6e41d72507f877).network"]]]
SecurityGroups              ["private.securitygroup"]
PositionAndCount            {
 "Item1":1,"Item2":1 }

Hypervisor                  nitro
TotalItemsCount             29
TotalFilteredItemsCount     29
Tags                        {
 "user":["test-user"] }


PS C:\Users\admin> $items[5].AdditionalData
Key                         Value
---                         -----
ServiceOfferingDescription T3 Small Instance
TenancyType                 Shared
HibernationSupported        False
```



```
NetworkMappings             [[["subnet-074242d7489b7209e","us-east-1a.
    availabilityzone\\10.0.128.0/17 (vpc-0fa6e41d72507f877).network"
    ]],[["subnet-074242d7489b7209e","us-east-1a.availabilityzone
    \\10.0.128.0/17 (vpc-0fa6e41d72507f877).network"]]]
SecurityGroups              ["private.securitygroup"]
PositionAndCount            {
 "Item1":2,"Item2":12 }

Hypervisor                  nitro

PS C:\Users\admin> $items = Get-HypInventoryItem -ResourceType "
    launchtemplateversion" -LiteralPath "XDHyp:\HostingUnits\ctx-test" -
    MaxRecords 200 -Template "ami-01548d6ca90670e7e" -Warn $true
PS C:\Users\admin> $items[3].AdditionalData
Key Value
--- -----
ServiceOfferingDescription T2 Medium Instance
TenancyType Shared
HibernationSupported False
PositionAndCount {
 "Item1":1,"Item2":1 }

Hypervisor xen
Warning Block device count mismatch;


PS C:\Users\admin>  Get-HypInventoryItem -ResourceType "
    launchtemplateversion" -LiteralPath "XDHyp:\HostingUnits\ctx-test" -
    MaxRecords 5 -Warn $true -AdditionalDataFilter "{
 'Hypervisor' : 'nitro' }
 " | Select Name
Name
----
lt-0bb652503d45dcbcd (9)
lt-0e1146edd2bbd3d18 (1)
lt-0e1146edd2bbd3d18 (2)
lt-0e1146edd2bbd3d18 (3)
lt-0e1146edd2bbd3d18 (4)
```

EXAMPLE 5

Examples of Additional Data return for snapshot data type with different filtering parameters. Note: Tags and Custom Attributes both are used in -Tags parameter. The first element in AdditionalData will list all possbile Tag values.

```
PS C:\Users\admin> $items = Get-HypInventoryItem -LiteralPath "XDHyp:\
    HostingUnits\ctx-test" -ResourceType "snapshot" -MaxRecords 10
PS C:\Users\admin> $items[0].AdditionalData

Key                     Value
---                     -----
```



```
GuestOsId                windows2022srvNext_64Guest
TotalItemsCount          47
TotalFilteredItemsCount 47
Tags                     {
 "Tag_1": ["Category_1]", "Custom_Attribute": ["
    Custom_Attribute_Value_1", "Custom_Attribute_Value_2"] }



PS C:\Users\admin> $items = Get-HypInventoryItem -LiteralPath 'XDHyp:\
    HostingUnits\ctx-test' -ResourceType "snapshot" -MaxRecords 10 -Tags
     '{
  "Tag_1": "Category_1", "Custom_Attribute": "Custom_Attribute_Value_1"
       }
 ' -AdditionalDataFilter '{
 "GuestOsId": "windows2022srvNext_64Guest" }
 ' | Select Name

Name
----
VMSnapshot20

PS C:\Users\admin> $items[0].AdditionalData

Key                      Value
---                      -----
GuestOsId                windows2022srvNext_64Guest
TotalItemsCount          47
TotalFilteredItemsCount 1
Tags                     {
 "Tag_1": ["Category_1]", "Custom_Attribute": ["
    Custom_Attribute_Value_1", "Custom_Attribute_Value_2"] }
```

Parameters

-LiteralPath

This specifies the XDHyp:\ path to the Hosting Unit for the inventory to use. This parameter is required.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	True

-ResourceType

This parameter specifies the type of the resources to return. This is a required parameter and corresponds to the resource Name extension, such as 'template', 'vm', 'launchtemplateversion'(only supported in AWS), or 'snapshot'(only supported in VMware).

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-ContinuationToken

This parameter controls where to start including items from the inventory. This is usually the 'Id' of an item and the included items begin after the given item is found.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-MaxRecords

The maximum number of items to return for the given call of the cmdlet. The default is to return one item.

Type:	Int32
Position:	Named
Default value:	1
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ContainsName

This parameter filters items so that only the ones containing the given string in the ‘Name‘ are included.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-ForwardDirection

Specifies whether the items are returned in forward page order (the default) or reverse.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Tags

This parameter specifies a JSON dictionary where the keys are the tag names and the values are the tag values. The parameter is used to filter items which have the given tags. An item must have all the given tags to match. Special values of ‘Not tagged‘ and ‘All values‘ are supported. For VMware, refer for additional usage details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-Id

This parameter returns a single item with the given Identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-AdditionalDataFilter

This specifies a JSON dictionary of parameters to use in filtering resources using the AdditionalData properties. All values given must match for an item to be included.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-Template

This parameter is used for the Machine Profile ResourceTypes only. It specifies a Template identifier or name to use in validation of the Machine Profile with the given Template. The validation will pass if the given Machine Profile resource is compatible with the given Template. Only Machine Profiles which pass this validation are included.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-Warn

This parameter is used for along with any validation filtering parameter (such as TemplateId). It specifies that rather than excluding the items which failed validation, the item is included but with a Warn-

ing property added to the AdditionalData. This property carries a string which describes the reason for validation failure.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Force

Specifies whether to return the full list of items or whether to exclude the items created as part of Machine Creation Services. The default behavior is to exclude the MCS created items.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

A list of inventory items which are found in the given Hosting Unit inventory of the given ResourceType and matching any optional filtering parameters. As a convenience the first item of the list contains AdditionalData fields for 'TotalItemsCount', 'TotalFilteredItemsCount' and 'Tags'. These fields provide the counts of all the items of the ResourceType in the inventory, the set of items which would match the filters and a set of tags which may be used to filter the items, respectively.

Notes

For filtering in VMware, Tag's name is treated as the key and its category name as the value. Apart from Tags, VMware also has Custom Attributes which are key-value pair based metadata for the resources. Both of them can be applied to filtering using '-Tags' parameter.

In VMware, snapshots don't have tags or custom attributes of their own. So, these values are fetched from their parent VMs.

Related Links

Get-HypLocalizedString

March 11, 2024

Obtains human-readable and locale-aware status messages from a hosting connection.

Syntax

```
Get-HypLocalizedString
    [-Locale] <String>
    [-Category] <String>
    [[-LookupKey] <String>]
    [-LiteralPath] <String>
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

TODO

Examples

EXAMPLE 1

This command illustrates how a single error/exception message can be obtained for a hosting operation that failed (such as a provisioning operation). The caller specifies that they need to display the message in the en-US locale. They also specify a lookup key or message ID for the message, which will have been obtained as an exception code from the failed operation. The locale and key/ID are traded for a readable message, which also includes substitutions.

```
Get-HypLocalizedString -LiteralPath XDHyp:\Connections\MyCloud -
    Category Exception -Locale "en-US" -LookupKey "MyCloud.
    InsufficientFreeIpSpace"

          Category             : Exception
          LookupKey            : MyCloud.InsufficientFreeIpSpace
          DisplayText          : There are not enough free IP addresses
              on the network {
 0 }
```

Parameters

-LiteralPath

Specifies the path to the hypervisor connection whose strings are being queried. The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>

or <drive>:\Connections{<Connection Uid>}

Type:	String
Position:	1
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Locale

The requested locale for the message, such as “en-US”.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Category

The category of the localized string.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LookupKey

The lookup key or message ID.

Type:	String
Position:	4

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Get-HypLocalizedString (LiteralPath parameter).

Outputs

Citrix.Host.Sdk.HypervisorString

Get-HypLocalizedString returns zero or more instances of the HypervisorString object, each of which contain the following properties:

Category <string>

Specifies the category of the string, such as “Exception”.

LookupKey <string>

Specifies the unique lookup key or message ID within the category.

DisplayText <string>

Specifies the locale-aware and human-readable text.

Notes

In the case of failure, the following errors can be produced.

Error Codes

ConnectionsPathInvalid

The path provided is not to an item in the Connections subdirectory of the host service provider.

HypervisorConnectionObjectNotFound

The path provided could not be resolved to an existing hypervisor connection. The name or GUID is invalid.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin

Get-HypPvsDiskInfo

March 11, 2024

Gets instances of Pvs Disk Info.

Syntax

```
Get-HypPvsDiskInfo
   -FarmId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-HypPvsDiskInfo
   [-FarmId <Guid>]
   -SiteId <Guid>
   -StoreId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-HypPvsDiskInfo
   [-FarmId <Guid>]
   -SiteId <Guid>
   -DiskLocatorId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-HypPvsDiskInfo
   -SiteId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieve Pvs Disk Info of the farm ID provided

Examples

EXAMPLE 1

This command lists the PVS disk Locators of a farm.

```
Get-HypPvsDiskInfo -FarmId 00000000-0000-0000-0000-000000000000
```


Parameters

-FarmId

Specifies a filter for the FarmId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SiteId

Specifies a filter for the SiteId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-StoreId

Specifies a filter for the StoreId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DiskLocatorId

Specifies a filter for the DiskLocatorId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

The GUID of a farm to retrieve from.

Guid

The GUID of a site to retrieve from.

Guid

The GUID of a site to retrieve from.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Register-HypPvsSite
- Unregister-HypPvsSite

Get-HypPvsServer

March 11, 2024

Gets instances of PvsServers.

Syntax

```
Get-HypPvsServer
    [-ServerId <Guid>]
    [-ServerName <String>]
    [-ServerAddress <String>]
    [-Property <String[]>]
```



```
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve PvsServer whose properties match the given filter criteria. If no parameters are specified, this cmdlet retrieves all PvsSite objects.

Examples

EXAMPLE 1

This command lists the PVS servers that match the corresponding filters.

```
Get-HypPvsServer -ServerId 00000000-0000-0000-0000-000000000000 -
    ServerName "exampleServer" -ServerAddress "xyz.abc.local"
```

Parameters

-ServerId

Specifies a filter for the ServerId.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ServerName

Specifies a filter for the ServerName.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ServerAddress

Specifies a filter for the ServerAddress.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

The GUID of a server to retrieve.

Outputs

Citrix.Host.Sdk.PvsServer

This cmdlet returns matching PvsServer objects.

Related Links

- Register-HypPvsServer
- Unregister-HypPvsServer

Get-HypPvsSite

March 11, 2024

Gets instances of PvsSites.

Syntax

```
Get-HypPvsSite
    [-SiteId <Guid>]
    [-SiteName <String>]
    [-FarmId <Guid>]
    [-FarmName <String>]
    [-ResourceLocation <Guid>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieve PvsSite whose properties match the given filter criteria. If no parameters are specified, this cmdlet retrieves all PvsSite objects.

Examples

EXAMPLE 1

This command lists the PVS sites that match the corresponding filters.

```
Get-HypPvsSite -SiteId 00000000-0000-0000-0000-000000000000 -SiteName "
    exampleSite" -FarmId 00000000-0000-0000-0000-000000000000 -FarmName
    "exampleFarm" -ResourceLocation 00000000-0000-0000-0000-000000000000
```


Parameters

-SiteId

Specifies a filter for the SiteId.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-SiteName

Specifies a filter for the SiteName.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-FarmId

Specifies a filter for the FarmId.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-FarmName

Specifies a filter for the FarmName.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ResourceLocation

Specifies a filter for the ResourceLocation.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

The GUID of a site to retrieve.

Outputs

Citrix.Host.Sdk.PvsSite

This cmdlet returns matching PvsSite objects.

Related Links

- Register-HypPvsSite
- Unregister-HypPvsSite

Get-HypPvsStore

March 11, 2024

Gets instances of Pvs Store.

Syntax

```
Get-HypPvsStore
   -FarmId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-HypPvsStore
   [-FarmId <Guid>]
   -SiteId <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Retrieve Pvs Disk Locators of the farm ID provided

Examples

EXAMPLE 1

This command lists the PVS stores of a farm.

```
Get-HypPvsStore -FarmId 00000000-0000-0000-0000-000000000000
```


Parameters

-FarmId

Specifies a filter for the FarmId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SiteId

Specifies a filter for the SiteId.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

The GUID of a farm to retrieve from.

Guid

The GUID of a site to retrieve from.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Register-HypPvsSite
- Unregister-HypPvsSite

Get-HypScopedObject

March 11, 2024

Gets the details of the scoped objects for the Host Service.

Syntax

```
Get-HypScopedObject
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-ObjectType <ScopedObjectType>]
    [-ObjectId <String>]
    [-ObjectName <String>]
    [-Description <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Examples

EXAMPLE 1

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.

```
Get-HypScopedObject -ObjectType Scheme

ScopeId     : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName   : Sales
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

```



```
ScopeId     : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName   : Finance
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     :
ScopeName   :
ObjectType  : Scheme
ObjectId    : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
ObjectName  : AnotherScheme
Description : Another scheme in no scopes
```

Parameters

-ScopeId

Gets scoped object entries for the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ScopeName

Gets scoped object entries with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectType

Gets scoped object entries for objects of the given type.

Type:	ScopedObjectType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ObjectId

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectName

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Gets scoped object entries for objects with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.ScopedObject

The Get-HypScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Specifies the unique identifier of the scope.

ScopeName <String>

Specifies the display name of the scope.

ObjectType <ScopedObjectType>

Type of the object this entry relates to.

ObjectId <String>

Unique identifier of the object.

ObjectName <String>

Display name of the object

Description <String>

Description of the object (possibly $null if the object type does not have a description).

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin

Get-HypServerAddressDetails

March 11, 2024

Gets all the available addresses for a hypervisor connection.

Syntax

```
Get-HypServerAddressDetails
   [-LiteralPath] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this cmdlet to retrieve all the available hypervisor connection addresses that can be used to connect to the same Server pool. When used in conjunction with Add-HypHypervisorAddress, you can easily populate a connection with all the addresses that can be used to provide full failover support for a Server connection.

If the addresses are https addresses, the command uses the certificates installed on the Servers to provide https addresses along with thumbprints and pem certs.

Examples

EXAMPLE 1

Gets the available addresses for the connection "MyConnection".

```
Get-HypServerAddressDetails -LiteralPath XDHyp:\Connections\
   MyConnection

https:\\myserver.com
https:\\myServer1.com
```


EXAMPLE 2

Adds all the available addresses for the Server pool in “MyConnection”to the hypervisor connection.

```
Get-HypServerAddressDetails -LiteralPath XDHyp:\Connections\
    MyConnection | Add-HypHypervisorConnectionAddress -LiteralPath XDHyp
    :\Connections\MyConnectionPath
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hypervisor connection item to which to add the address. The path specified must be in one of the following formats:

<drive>:\Connections\<HypervisorConnectionName>

or <drive>:\Connections{HHypervisorConnection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

HypervisorHostDetails

Get-HypServerAddressDetails returns a list of strings containing the available address and their certs/tumbprint info.

Notes

For this to work as required with https connections, the certificates installed on the Servers must be trusted by all controllers. Typically this means having the root certificate for the certificate trust chain installed on all controllers.

In the case of failure, the following errors can result.

Error Codes

InputConnectionsPathInvalid

The path provided is not to an item in a sub-directory of a hosting unit item.

SupportsEnumerateHypervisorHostDetailsCapabilityMissing

The hypervisor is missing EnumerateHypervisorHostDetailsCapability.

HypervisorConnectionObjectNotFound

The hypervisor connection at the path specified cannot be located.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the Desktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionAddress

Get-HypService

March 11, 2024

Gets the service record entries for the Host Service.

Syntax

```
Get-HypService
   [-Metadata <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns instances of the Host Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Host Service running in the current service group.

```
Get-HypService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.Service

The Get-HypServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin

Get-HypServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Host Service on the controller.

Syntax

```
Get-HypServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Host Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Host Service.

```
Get-HypServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin

Get-HypServiceInstance

March 11, 2024

Gets the service instance entries for the Host Service.

Syntax

```
Get-HypServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns service interfaces published by instances of the Host Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Host Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
1  Get-HypServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.ServiceInstance

The Get-HypServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Hyp.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Host.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypServiceStatus
- Reset-HypServiceGroupMembership

Get-HypServiceStatus

March 11, 2024

Gets the current state of the Host Service on the controller.

Syntax

```
Get-HypServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the Host Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Host Service.

```
Get-HypServiceStatus
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-HypServiceStatus command returns an object containing the status of the Host Service together with extra diagnostics information.

DBUnconfigured

The Host Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Host Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Host Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Host is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Host is connected to a database that is valid, but it does not have the full functionality required so the Host cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Host Service currently in use is incompatible with the version of the Host Service schema on the database. Upgrade the Host Service to a more recent version.

DBOlderVersionThanService

The version of the Host Service schema on the database is incompatible with the version of the Host Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Host Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Host Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Host and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Host service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypDBConnection
- Test-HypDBConnection
- Get-HypDBConnection
- Get-HypDBSchema

Get-HypVMMacAddress

March 11, 2024

Retrieves a list the MAC addresses for the VMs in the specified connection.

Syntax

```
Get-HypVMMacAddress
    [-LiteralPath] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this command to obtain a list of MAC addresses of all the virtual machines in the specified connection.

Examples

EXAMPLE 1

This command gets the MAC addresses for the connection called "MyConnection".

```
Get-HypVMMacAddress -LiteralPath XDHyp:\Connections\MyConnection

MacAddress                                VMId
----------                                ----
52:b0:1c:ed:60:fa                         f3395d2a-a196-41c2-e37d-764
    acf871599
62:6f:f0:40:d5:af                         5f4457b0-cc3c-f806-8ca7-5
    f57e4bdf2d1
4e:a5:9f:00:b2:0c                         3115177b-85a9-d8ee-d0f9-0
    c7437483c09
```

EXAMPLE 2

This command gets the MAC addresses for the connection at the current directory. The dot (.) represents the current location (not its contents).

```
XDHyp:\Connections\MyConnection>Get-HypVMMacAddress -Path .

MacAddress                                VMId
----------                                ----
52:b0:1c:ed:60:fa                         f3395d2a-a196-41c2-e37d-764
    acf871599
62:6f:f0:40:d5:af                         5f4457b0-cc3c-f806-8ca7-5
    f57e4bdf2d1
4e:a5:9f:00:b2:0c                         3115177b-85a9-d8ee-d0f9-0
    c7437483c09
```


EXAMPLE 3

This command gets the MAC addresses for the connection that has a ConnectionUid of 268c66db-9b8c-47f6-9265-42326dbff006.

```
Get-HypVMMacAddress -LiteralPath Xdhyp:\connections\"{
 268c66db-9b8c-47f6-9265-42326dbff006 }
 "
```

Parameters

-LiteralPath

The path to a connection item in the hosting provider. Paths to anything other than a connection item will result in an error being returned. The path can be provided as either

<drive>:\connections\<Connection Name>

or <drive:>\connections{<Connection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Get-HypConfigurationDataForItem

Outputs

Citrix.Host.Sdk.HypervisorVMObject

Get-HypVMMMacAddress returns an object containing the following properties.

MacAddress <string> - specifies the MAC address of the VM. VMId <string> - specifies the identifier for the VM as defined in the hypervisor hosting it.

Notes

The path must refer to a connection item. Hosting unit items are not valid.

In the case of failure, the following errors can result.

Error Codes

InputConnectionsPathInvalid

If the path is not provided in an expected format, an InputConnectionsPathInvalid error results.

HypervisorConnectionObjectNotFound

The hypervisor connection object specified cannot be found.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-Item

Get-HypVolumeServiceConfiguration

March 11, 2024

Gets instances of the VolumeServiceConfiguration that are configured for this site.

Syntax

```
Get-HypVolumeServiceConfiguration
   [-VolumeServiceConfigurationUid <Guid>]
   [[-VolumeServiceConfigurationName] <String>]
   [-ConnectionType <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Retrieve VolumeServiceConfigurations whose properties match the given filter criteria. If no parameters are specified, this cmdlet retrieves all VolumeServiceConfiguration objects.

Examples

EXAMPLE 1

This command lists the site default volume service configuration for connections that use Citrix Cloud Platform.

```
Get-HypVolumeServiceConfiguration -VolumeServiceConfigurationName
    SiteDefault -ConnectionType CloudPlatform
```

Parameters

-VolumeServiceConfigurationName

Specifies a filter for the configuration name.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-VolumeServiceConfigurationUid

Specifies a filter for the configuration ID.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ConnectionType

Specifies a filter for the cloud connection type, such as “AWS”or “CloudPlatform”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Hyp_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Hyp_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The name of a configuration (or set of configurations) to retrieve.

Outputs

Citrix.Host.Sdk.VolumeServiceConfiguration

This cmdlet returns matching VolumeServiceConfiguration objects.

Related Links

- Set-HypVolumeServiceConfiguration

Get-HypXenServerAddress

March 11, 2024

Gets all the available addresses for a XenServer hypervisor connection.

Syntax

```
Get-HypXenServerAddress
    [-LiteralPath] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this cmdlet to retrieve all the available hypervisor connection addresses that can be used to connect to the same XenServer pool. When used in conjunction with Add-HypHypervisorAddress, you can easily populate a connection with all the addresses that can be used to provide full failover support for a XenServer connection.

If the addresses are https addresses, the command uses the certificates installed on the XenServers to provide suitable https addresses where possible. Only servers that can be resolved are returned.

Examples

EXAMPLE 1

Gets the available addresses for the connection "MyConnection".

```
Get-HypXenServerAddress -LiteralPath XDHyp:\Connections\MyConnection

https:\\myserver.com
https:\\myServer1.com
```

EXAMPLE 2

Adds all the available addresses for the XenServer pool in "MyConnection"to the hypervisor connection.

```
Get-HypXenServerAddress -LiteralPath XDHyp:\Connections\MyConnection |
    Add-HypHypervisorConnectionAddress -LiteralPath XDHyp:\Connections\
    MyConnectionPath
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hypervisor connection item to which to add the address. The path specified must be in one of the following formats:

<drive>:\Connections\<HypervisorConnectionName>

or <drive>:\Connections{HHypervisorConnection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

Get-HypXenServerAddress returns a list of strings containing the available address.

Notes

For this to work as required with https connections, the certificates installed on the XenServers must be trusted by all controllers. Typically this means having the root certificate for the certificate trust chain installed on all controllers. Wildcard certificates are not supported for this operation.

In the case of failure, the following errors can result.

Error Codes

InputConnectionsPathInvalid

The path provided is not to an item in a sub-directory of a hosting unit item.

HypervisorConnectionNotXenServer

The hypervisor connection to which the path refers is not for XenServer.

HypervisorConnectionObjectNotFound

The hypervisor connection at the path specified cannot be located.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionAddress

Grant-HypSecurityGroupEgress

March 11, 2024

Adds an egress rule to a security group.

Syntax

```
Grant-HypSecurityGroupEgress
      [-LiteralPath] <String>
      -Protocol <String>
      [-FromPort <Decimal>]
      [-ToPort <Decimal>]
      -GroupId <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Grant-HypSecurityGroupEgress
      [-LiteralPath] <String>
      -Protocol <String>
      [-FromPort <Decimal>]
      [-ToPort <Decimal>]
      -IPRange <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet is deprecated. Adding an egress rule permits network traffic from instances within the security group to pass to one or more destination CIDR IP address ranges or security groups.

Examples

EXAMPLE 1

Create a security group and grant full egress to anywhere.

```
$Group = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    -Name MySecurityGroup -Description 'Example group'
Grant-HypSecurityGroupEgress $Group.FullPath -Protocol '-1' -IPRange '
    0.0.0.0/16'
```

EXAMPLE 2

Make 2 security groups and permit group 1 to ping group 2.

```
$Group1 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
     -Name MySecurityGroup1 -Description 'Example group 1'
$Group2 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    \MySecurityGroup2 -Description 'Example group 2'
Grant-HypSecurityGroupEgress $Group1.FullPath -FromPort 0 -ToPort 0 -
    Protocol icmp -GroupId $Group2.Id
Grant-HypSecurityGroupIngress $Group2.FullPath -FromPort 0 -ToPort 0 -
    Protocol icmp -GroupId $Group1.Id
```

Parameters

-LiteralPath

Specifies the full XDHyp provider path to the security group, equivalent to the FullPath property of the security group object. The path can specify a security group relative to a hypervisor conection or hosting unit.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Protocol

Specifies the protocol name or number. Protocol numbers can be found at: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

Use -1 to specify all protocols.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-GroupId

Specifies one or more destination security groups to which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IPRange

Specifies one or more destination CIDR IP address ranges to which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FromPort

The start of the port range for port based protocols. For ICMP this specifies the type number.

Use -1 to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ToPort

The end of the port range for port based protocols. For ICMP this specifies the type number, where -1 can be used to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The LiteralPath can be piped in.

Outputs

None

By default, this cmdlet returns no output.

Notes

Security groups can be added and removed using the New-Item and Remove-Item cmdlets.

Related Links

- Amazon AuthorizeSecurityGroupEgress
- IANA protocol numbers
- Grant-HypSecurityGroupIngress
- Revoke-HypSecurityGroupIngress
- Revoke-HypSecurityGroupEgress

Grant-HypSecurityGroupIngress

March 11, 2024

Adds an ingress rule to a security group.

Syntax

```
Grant-HypSecurityGroupIngress
      [-LiteralPath] <String>
      -Protocol <String>
      [-FromPort <Decimal>]
      [-ToPort <Decimal>]
      -GroupId <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Grant-HypSecurityGroupIngress
      [-LiteralPath] <String>
      -Protocol <String>
      [-FromPort <Decimal>]
      [-ToPort <Decimal>]
      -IPRange <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet is deprecated. Adding an egress rule permits network traffic from source CIDR IP address ranges or security groups to pass to instances within a security group.

Examples

EXAMPLE 1

Create a security group and grant ingress on port 80 from anywhere.

```
$Group = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    -Name MySecurityGroup -Description 'Example group'
Grant-HypSecurityGroupIngress $Group.FullPath -FromPort 80 -ToPort 80 -
    Protocol tcp -IPRange '0.0.0.0/0'
```

EXAMPLE 2

Make 2 security groups and permit group 1 access to group 2 only on port 8080 while granting full access to group 1 from group 2.

```
$Group1 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
     -Name MySecurityGroup1 -Description 'Example group 1'
$Group2 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    \MySecurityGroup2 -Description 'Example group 2'
Grant-HypSecurityGroupEgress $Group1.FullPath -FromPort 8080 -ToPort
    8080 -Protocol tcp -GroupId $Group2.Id
Grant-HypSecurityGroupIngress $Group2.FullPath -FromPort 8080 -ToPort
    8080 -Protocol tcp -GroupId $Group1.Id
Grant-HypSecurityGroupEgress $Group2.FullPath -Protocol '-1' -GroupId
    $Group1.Id
Grant-HypSecurityGroupIngress $Group1.FullPath -Protocol '-1' -GroupId
    $Group2.Id
```

Parameters

-LiteralPath

Specifies the full XDHyp provider path to the security group, equivalent to the FullPath property of the security group object. The path can specify a security group relative to a hypervisor conection or hosting unit.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Protocol

Specifies the protocol name or number. Protocol numbers can be found at: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

Use -1 to specify all protocols.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-GroupId

Specifies one or more source security groups from which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IPRange

Specifies one or more source CIDR IP address ranges from which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FromPort

The start of the port range for port based protocols. For ICMP this specifies the type number.

Use -1 to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ToPort

The end of the port range for port based protocols. For ICMP this specifies the type number, where -1 can be used to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The LiteralPath can be piped in.

Outputs

None

By default, this cmdlet returns no output.

Notes

Security groups can be added and removed using the New-Item and Remove-Item cmdlets.

Related Links

- Amazon AuthorizeSecurityGroupEgress
- IANA protocol numbers
- Grant-HypSecurityGroupEgress
- Revoke-HypSecurityGroupIngress
- Revoke-HypSecurityGroupEgress

New-HypStorage

March 11, 2024

Creates a new storage tier definition for use in a subsequent New-Item operation.

Syntax

```
New-HypStorage
   [-JobGroup] <Guid>
   -StorageType <StorageType>
   -StoragePath <String[]>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to create a new storage tier definition, this definition can then be associated with a subsequent New-Item operation via the JobGroup reference.

Examples

EXAMPLE 1

The command creates a new job group with the unique id specified by $job which can subsequently be fed into the New-Item cmdlet. It defines a TemporaryStorage tier which will cause disks to be added to the storage path 'XDHyp:\Connections\MyConnection\Primary Storage.storage'.


```
$job = [Guid]::NewGuid()
New-HypStorage -JobGroup $job -StorageType TemporaryStorage -
    StoragePath @('XDHyp:\Connections\MyConnection\Primary Storage.
    storage')
```

Parameters

-JobGroup

Specifies the JobGroup uid that is used to associate data from calls to this cmdlet with the subsequent New-Item operation.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StorageType

The type of the new storage tier. Currently the only storage type is TemporaryStorage.

Type:	StorageType
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StoragePath

Specifies the path to the storage that will be added. The path must be in one of the following formats: <drive>:\Connections\<ConnectionName>\MyStorage.storage or <drive>:\Connections{<ConnectionUid>}\MyStorage.storage

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_HypHostSnapin
- about_HypStorage

New-HypVMSnapshot

March 11, 2024

Creates a new snapshot for the specified VM item path.

Syntax

```
New-HypVMSnapshot
    [-SnapshotName] <String>
    [[-SnapshotDescription] <String>]
    [-LoggingId <Guid>]
    [-LiteralPath] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this command to create a new snapshot of a virtual machine, for a given Host Service provider path to a VM. The resulting snapshot can then be used for operations that require a snapshot to work.

Examples

EXAMPLE 1

This command creates a snapshot of a VM called 'MyVm.vm'within a hypervisor connection called 'MyConnection'.

```
New-HypVMSnapshot -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm
    -SnapshotName "New snapshot" -SnapshotDescription "Example snapshot"

                    XDHyp:\Connections\MyConnection\MyVm.vm\New
                       snapshot.snapshot
```

Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the virtual machine item for which to create a new snapshot. The path specified must be in one of the following formats:

<drive>:\Connections\<Connection Name>\<Item Path of VM object>

or <drive>:\Connections{<connection Uid>\<Item Path of VM object>}

or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object>

or <drive>:\HostingUnits{<hostingUnit Uid>\<Item Path of VM object>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SnapshotName

The name of the new snapshot. This is visible in the hypervisor management console.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SnapshotDescription

The description to add to the snapshot. This is visible in the hypervisor management console.

Type:	String
Position:	3
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Get-HypConfigurationDataForItem

Outputs

String

The provider path to the newly created snapshot.

Notes

In the case of failure, the following errors can result.

Error Codes

InputHypervisorItemPathInvalid

The path provided is not to an item in a sub-directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

SnapshotNameAlreadyInUse

The specified name is already in use and will cause a name resolution clash.

FailedToCreateSnapshot

The snapshot creation process failed.

HypervisorInMaintenanceMode

The hypervisor is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

SnapshotChainTooLong

Snapshot creation failed. Snapshot chain is too long.

SnapshotCreationNotAuthorized

Snapshot creation failed. User not authorized to create snapshots.

Related Links

- about_HypHostSnapin

Register-HypPvsServer

March 11, 2024

Register PVS Site.

Syntax

```
Register-HypPvsServer
        -SiteId <Guid>
        -SiteName <String>
        -FarmId <Guid>
        -FarmName <String>
        -ServerId <Guid>
        -ServerName <String>
        -ServerAddress <String>
        -Thumbprint <String>
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```


Description

Use this command to register PvsSites with with SiteId FarmId and ResourceLocation.

Examples

EXAMPLE 1

This command registers a PvsServer with 00000000-0000-0000-0000-000000000000 in each attribute

```
Register-HypPvsServer -SiteId 00000000-0000-0000-0000-000000000000 -
    SiteName exampleSiteName -FarmId
    00000000-0000-0000-0000-000000000000 -FarmName exampleFarmName -
    ServerId 00000000-0000-0000-0000-000000000000 -ServerAddress xyz.abc
    .local -ServerName exampleServerName
```

Parameters

-SiteId

The ID of the PVS site.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SiteName

The Name of the PVS site.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-FarmId

The ID of the PVS farm.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-FarmName

The Name of the PVS farm.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ServerId

The Server ID of the PVS server.

Type:	Guid
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ServerName

The PVS Server Name

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ServerAddress

The PVS Server Address

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Thumbprint

The PVS Server Certificate SSL Thumbprint

Type:	String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- Unregister-HypPvsServer

Register-HypPvsSite

March 11, 2024

Register PVS Site.

Syntax

```
Register-HypPvsSite
        -SiteId <Guid>
        -SiteName <String>
        -FarmId <Guid>
        -FarmName <String>
        -ResourceLocation <Guid>
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```


Description

Use this command to register PvsSites with with SiteId FarmId and ResourceLocation.

Examples

EXAMPLE 1

This command registers a PvsSite with 00000000-0000-0000-0000-000000000000 in each attribute

```
Register-HypPvsSite -SiteId 00000000-0000-0000-0000-000000000000 -
    SiteName exampleSiteName -FarmId
    00000000-0000-0000-0000-000000000000 -FarmName exampleFarmName -
    ResourceLocation 00000000-0000-0000-0000-000000000000
```

Parameters

-SiteId

The ID of the PVS site.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-SiteName

The Name of the PVS site.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-FarmId

The ID of the PVS farm.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-FarmName

The Name of the PVS farm.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ResourceLocation

The ResourceLocation ID of the PVS server.

Type:	Guid
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- Get-HypPvsSite
- Unregister-HypPvsSite

Remove-HypHostingUnitMetadata

March 11, 2024

Removes metadata from the given HostingUnit.

Syntax

```
Remove-HypHostingUnitMetadata
       [-HostingUnitUid] <Guid>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-HypHostingUnitMetadata
       [-HostingUnitUid] <Guid>
       -Map <PSObject>
       [-LoggingId <Guid>]
```



```
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-HypHostingUnitMetadata
       [-HostingUnitName] <String>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-HypHostingUnitMetadata
       [-HostingUnitName] <String>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-HypHostingUnitMetadata
       [-InputObject] <HostingUnit[]>
       -Name <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-HypHostingUnitMetadata
       [-InputObject] <HostingUnit[]>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given HostingUnit.

Examples

EXAMPLE 1

Remove all metadata from all HostingUnit objects.

```
Get-HypHostingUnit | % {
  Remove-HypHostingUnitMetadata -Map $_.MetadataMap  }
```


Parameters

-HostingUnitUid

Id of the HostingUnit

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HostingUnitName

Name of the HostingUnit

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HostingUnit[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHostingUnitMetadata
- Set-HypHostingUnitMetadata

Remove-HypHostingUnitNetwork

March 11, 2024

Removes networks from a hosting unit.

Syntax

```
Remove-HypHostingUnitNetwork
      [-NetworkPath] <String>
      [-LiteralPath] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Use this command to remove networks from a hosting unit. This does not remove the network from the hypervisor, only the reference to the network for the Host Service. After it is removed, the network is no longer available for associating with virtual NICs (when creating new virtual machines with the Machine Creation Service). Any virtual machines that have been created already continue to use the network until they are removed from the deployment. This command cannot be used if the connection for the hosting unit is in maintenance mode. If the network to be removed no longer exists on the hypervisor for the hosting unit, you must supply a fully qualified path to the network location.

Examples

EXAMPLE 1

The command removes the network called “newNetwork.network”from the hosting unit called “MyHostingUnit”

```
Remove-HypHostingUnitNetwork -LiteralPath XDHyp:\HostingUnits\
   MyHostingUnit -NetworkPath 'XDHyp:\HostingUnits\MyHostingUnits\
   newNetwork.network'
```

Parameters

-LiteralPath

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkPath

Specifies the path in the hosting unit provider of the network to remove. The path specified must be in one of the following formats: <drive>:\Connections\<HostingUnitName>\MyNetwork.network or <drive>:\Connections{<hostingUnit Uid>}\MyNetwork.network

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Remove-HypHostingUnitNetwork (Path parameter).

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitNetworkObjectToDeleteDoesNotExist

The network path specified is not part of the hosting unit.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHostingUnitNetwork

Remove-HypHostingUnitStorage

March 11, 2024

Removes storage from a hosting unit.

Syntax


```
Remove-HypHostingUnitStorage
      [-StoragePath <String>]
      [-StorageType <StorageType>]
      [-LiteralPath] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Use this command to remove storage locations from a hosting unit. This does not remove the storage from the hypervisor, only the reference to the storage for the Host Service. After removal, the storage is no longer used to store hard disks (when creating new virtual machines with the Machine Creation Service). The hard disks located already on the storage remain in place and virtual machines that have been created already continue to use the storage until they are removed from the deployment. Do not use this command if the connection for the hosting unit is in maintenance mode. If the storage location to be removed no longer exists on the hypervisor for the hosting unit, you must supply a fully qualified path to the storage location.

Examples

EXAMPLE 1

The command removes the OS storage location called "newStorage.storage"from the hosting unit called "MyHostingUnit"

```
Remove-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\
   MyHostingUnit -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\
   newStorage.storage'
```

EXAMPLE 2

The command removes all OS storage from all hosting units called "Host1".

```
Get-ChildItem XDHYP:\HostingUnits\Host\*.storage | Remove-
   HypHostingUnitStorage XDHYP:\HostingUnits\Host1
```

EXAMPLE 3

The command removes all PersonalvDisk storage from all hosting units called "Host1".


```
Get-ChildItem XDHYP:\HostingUnits\Host\*.storage | Remove-
    HypHostingUnitStorage -StorageType PersonalvDiskStorage
```

Parameters

-LiteralPath

Specifies the path to the hosting unit which defines the storage. The path must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StoragePath

Specifies the path in the hosting unit provider of the storage to remove. If StoragePath is not specified, all storage is removed from the hosting unit.

The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>\MyStorage.storage

or <drive>:\Connections{<Connection Uid>}\MyStorage.storage

or <drive>:\HostingUnits\<HostingUnitName>\MyStorage.storage

or <drive>:\HostingUnits{<hostingUnit Uid>}\MyStorage.storage

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-StorageType

Specifies the type of storage in StoragePath. Supported storage types are:

OSStorage

PersonalvDiskStorage

TemporaryStorage

Type:	StorageType
Position:	Named
Default value:	OSStorage
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Remove-HypHostingUnitStorage (StoragePath parameter).

Outputs

Citrix.XDPowerShell.HostingUnit

Remove-HypHostingUnitStorage returns an object containing the new definition of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.XDPowerShell.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use.

PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether or not the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Service.

NetworkId <string>

The hypervisor's internal identifier that represents the network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the network specified for the hosting unit.

AdditionalStorage

The list of additional storage items that the hosting unit can use.

Metadata <Citrix.XDPowerShell.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

Notes

After storage is removed, it is the administrator's responsibility to maintain its contents. The Citrix XenDesktop Machine Creation Service does not attempt to clean up any data that is stored in the storage location.

If all storage is removed from the hosting unit, other features of the Machine Creation Service stops functioning until some storage is added again.

In the case of failure, the following errors can result.

Error Codes

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitStorageObjectToDeleteDoesNotExist

The storage path specified is not part of the hosting unit.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHostingUnitStorage
- Set-HypHostingUnitStorage
- New-Item
- Add-HypMetadata

Remove-HypHypervisorConnectionAddress

March 11, 2024

Removes addresses from the list of available connection addresses.

Syntax

```
Remove-HypHypervisorConnectionAddress
      [-LiteralPath] <String>
      [-HypervisorAddress] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Use this command to remove addresses that can be used to connect to the hypervisor specified by the hypervisor connection. If all addresses are removed, the connection cannot be used until a valid address is added to the hypervisor connection.

Examples

EXAMPLE 1

The command removes the address “http:\\myserver.com”from the hypervisor connection called “MyHypervisorConnection”.

```
Remove-HypHypervisorConnectionAddress -LiteralPath XDHyp:\HostingUnits\
   MyHypervisorConnection -HypervisorAddress 'http:\\myserver.com'
```

EXAMPLE 2

The command removes all addresses from all the hypervisor connections currently defined.

```
Get-Item -Path XDHYP:\Connections\* | Remove-
   HypHypervisorConnectionAddress
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hosting unit item to which to add the address. The path specified must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{HostingUnit Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HypervisorAddress

Specifies the address to be removed. If this parameter is not provided, all addresses are removed from the hypervisor connection.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Remove-HypHypervisorConnectionAddress (Path parameter).

Outputs

None

By default, this cmdlet returns no output.

Notes

Changes to a hypervisor connection affect all entities that reference the connection. If all addresses are removed from the connection, any other entities that reference the hypervisor connection (e.g. hosting units) cannot make new connections to the hypervisor.

In the case of failure, the following errors can result.

Error Codes

InputConnectionsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HypervisorConnectionAddressForeignKeyObjectDoesNotExist

The hypervisor connection to which the address is to be added could not be located.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionAddress

Remove-HypHypervisorConnectionMetadata

March 11, 2024

Removes metadata from the given HypervisorConnection.

Syntax

```
Remove-HypHypervisorConnectionMetadata
      [-HypervisorConnectionUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionMetadata
      [-HypervisorConnectionUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionMetadata
      [-HypervisorConnectionName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionMetadata
      [-HypervisorConnectionName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionMetadata
      [-InputObject] <HypervisorConnection[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionMetadata
      [-InputObject] <HypervisorConnection[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given HypervisorConnection.

Examples

EXAMPLE 1

Remove all metadata from all HypervisorConnection objects.

```
Get-HypHypervisorConnection | % {
  Remove-HypHypervisorConnectionMetadata -Map $_.MetadataMap  }
```

Parameters

-HypervisorConnectionUid

Id of the HypervisorConnection

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionName

Name of the HypervisorConnection

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionMetadata
- Set-HypHypervisorConnectionMetadata

Remove-HypHypervisorConnectionScope

March 11, 2024

Remove the specified HypervisorConnection(s) from the given scope(s).

Syntax

```
Remove-HypHypervisorConnectionScope
      [-Scope] <String[]>
      -InputObject <HypervisorConnection[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionScope
      [-Scope] <String[]>
      -HypervisorConnectionUid <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypHypervisorConnectionScope
      [-Scope] <String[]>
      -HypervisorConnectionName <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-HypHypervisorConnectionScope command is used to remove one or more Hypervisor-Connection objects from the given scope(s).

There are multiple parameter sets for this cmdlet, allowing you to identify the HypervisorConnection objects in different ways:

- HypervisorConnection objects can be piped in or specified by the InputObject parameter
- The HypervisorConnectionUid parameter specifies objects by HypervisorConnectionUid
- The HypervisorConnectionName parameter specifies objects by HypervisorConnectionName (supports wildcards)

To remove a HypervisorConnection from a scope you need permission to change the scopes of the HypervisorConnection.

If the HypervisorConnection is not in a specified scope, that scope will be silently ignored.

Examples

EXAMPLE 1

Removes a single HypervisorConnection from the 'Finance'scope.

```
Remove-HypHypervisorConnectionScope Finance -HypervisorConnectionUid
    6702C5D0-C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 2

Removes a single HypervisorConnection from multiple scopes.

```
Remove-HypHypervisorConnectionScope Finance,Marketing -
    HypervisorConnectionUid 6702C5D0-C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 3

Removes all visible HypervisorConnection objects from the 'Finance'scope.

```
Get-HypHypervisorConnection | Remove-HypHypervisorConnectionScope
    Finance
```

EXAMPLE 4

Removes HypervisorConnection objects with a name starting with an 'A'from the 'Finance'scope.

```
Remove-HypHypervisorConnectionScope Finance -HypervisorConnectionName A
    *
```


Parameters

-Scope

Specifies the scopes to remove the objects from.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

Specifies the HypervisorConnection objects to be removed.

Type:	HypervisorConnection[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionUid

Specifies the HypervisorConnection objects to be removed by HypervisorConnectionUid.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	False

-HypervisorConnectionName

Specifies the HypervisorConnection objects to be removed by HypervisorConnectionName.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionScope
- Get-HypScopedObject

Remove-HypMetadata

March 11, 2024

Removes metadata from a hypervisor connection or hosting unit.

Syntax

```
Remove-HypMetadata
      [-Property <String>]
      [-LiteralPath] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Use this command to remove custom data from a specific hosting unit or hypervisor connection. If the Property parameter is not provided, all metadata is removed from the specified item.

Examples

EXAMPLE 1

The command removes the metadata with the property "MyProperty"from the hypervisor connection called "MyConnection".

```
Remove-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection -
    Property MyProperty
```

EXAMPLE 2

The command removes all the metadata from the hypervisor connection called "MyConnection".

```
Remove-HypMetadata -LiteralPath XDHyp:\Connections\MyConnection
```

EXAMPLE 3

The command removes all the metadata from all the hypervisor connections.

```
dir XDHyp:\connections | Remove-HypMetadata
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hosting unit or hypervisor connection item from which to remove the metadata. The path specified must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

or <drive>:\Connections\<Connection Name>

or <drive>:\Connections{<Connection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Property

Specifies the property name of the metadata to be removed.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Remove-HypMetadata (Path parameter).

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

InvalidPath

The path provided is not in the required format.

HostingUnitMetadataObjectToDeleteDoesNotExist

The hosting unit supplied in the path does not exist.

HypervisorConnectionObjectToDeleteDoesNotExist

The hypervisor connection supplied in the path does not exist.

MetadataContainerUndefined

The specified path does not reference a hosting unit or a hypervisor connection.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Remove-HypMetadata

Remove-HypServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax


```
Remove-HypServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-HypServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-HypService | % {
  Remove-HypServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Set-HypServiceMetadata

Reset-HypEnabledFeatureList

March 11, 2024

Refreshes the Host service's list of enabled features.

Syntax

```
Reset-HypEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Host Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Host service instance's list of enabled features.

```
1  Reset-HypEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-HypServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Host Service.

Syntax

```
Reset-HypServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Host Service access permissions and configuration service locations. The Reset-HypServiceGroupMembership command must be run on at least one instance of the service type (Hyp) after installation and registration with the configuration service. Without this operation, the Host services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-HypServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
   HypServiceGroupMembership
```


EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-HypServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Host.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-HypServiceGroupMembership command.

Outputs

Citrix.Host.Sdk.ServiceInstance

Reset-HypServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Host Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypServiceInstance
- Get-HypServiceStatus

Revoke-HypSecurityGroupEgress

March 11, 2024

Removes an egress rule from a security group.

Syntax

```
Revoke-HypSecurityGroupEgress
       [-LiteralPath] <String>
       -Protocol <String>
       [-FromPort <Decimal>]
```



```
        [-ToPort <Decimal>]
        -GroupId <String[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Revoke-HypSecurityGroupEgress
        [-LiteralPath] <String>
        -Protocol <String>
        [-FromPort <Decimal>]
        [-ToPort <Decimal>]
        -IPRange <String[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This cmdlet is deprecated. To remove a rule, specify parameters matching an existing rule's values.

Examples

EXAMPLE 1

Create a security group, grant full egress to anywhere, then revoke access and delete the security group.

```
$Group = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    -Name MySecurityGroup -Description 'Example group'
Grant-HypSecurityGroupEgress $Group.FullPath -Protocol '-1' -IPRange '
    0.0.0.0/0'
Revoke-HypSecurityGroupEgress $Group.FullPath -Protocol '-1' -IPRange '
    0.0.0.0/0'
Remove-Item $Group.FullPath
```

Parameters

-LiteralPath

Specifies the full XDHyp provider path to the security group, equivalent to the FullPath property of the security group object. The path can specify a security group relative to a hypervisor conection or hosting unit.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Protocol

Specifies the protocol name or number. Protocol numbers can be found at: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

Use -1 to specify all protocols.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-GroupId

Specifies one or more destination security groups to which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IPRange

Specifies one or more destination CIDR IP address ranges to which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FromPort

The start of the port range for port based protocols. For ICMP this specifies the type number.

Use -1 to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ToPort

The end of the port range for port based protocols. For ICMP this specifies the type number, where -1 can be used to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The LiteralPath can be piped in.

Outputs

None

By default, this cmdlet returns no output.

Notes

Security groups cannot be removed in AWS if they are referened by rules from other security groups.

Security groups can be added and removed using the New-Item and Remove-Item cmdlets.

Related Links

- Amazon AuthorizeSecurityGroupEgress
- IANA protocol numbers
- Grant-HypSecurityGroupIngress
- Grant-HypSecurityGroupEgress
- Revoke-HypSecurityGroupIngress

Revoke-HypSecurityGroupIngress

March 11, 2024

Removes an ingress rule from a security group.

Syntax

```
Revoke-HypSecurityGroupIngress
       [-LiteralPath] <String>
       -Protocol <String>
       [-FromPort <Decimal>]
       [-ToPort <Decimal>]
       -GroupId <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Revoke-HypSecurityGroupIngress
       [-LiteralPath] <String>
       -Protocol <String>
```



```
        [-FromPort <Decimal>]
        [-ToPort <Decimal>]
        -IPRange <String[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This cmdlet is deprecated. To remove a rule, specify parameters matching an existing rule's values.

Examples

EXAMPLE 1

Create 2 security groups, grant access from group 1 to group 2, then revoke access.

```
$Group1 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    -Name MySecurityGroup1 -Description 'Example group 1'
$Group2 = New-Item -ItemType SecurityGroup -Path XDHyp:\Connections\AWS
    -Name MySecurityGroup2 -Description 'Example group 2'
Grant-HypSecurityGroupEgress $Group1.FullPath -FromPort 8080 -ToPort
    8085 -Protocol tcp -GroupId $Group2.Id
Grant-HypSecurityGroupIngress $Group2.FullPath -FromPort 8080 -ToPort
    8085 -Protocol tcp -GroupId $Group1.Id
Revoke-HypSecurityGroupEgress $Group1.FullPath -FromPort 8080 -ToPort
    8085 -Protocol tcp -GroupId $Group2.Id
Revoke-HypSecurityGroupIngress $Group2.FullPath -FromPort 8080 -ToPort
    8085 -Protocol tcp -GroupId $Group1.Id
```

Parameters

-LiteralPath

Specifies the full XDHyp provider path to the security group, equivalent to the FullPath property of the security group object. The path can specify a security group relative to a hypervisor conection or hosting unit.

Type:	String
Position:	1
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Protocol

Specifies the protocol name or number. Protocol numbers can be found at: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml

Use -1 to specify all protocols.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-GroupId

Specifies one or more source security groups from which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IPRange

Specifies one or more source CIDR IP address ranges from which traffic will be permitted by this rule. This parameter cannot be specified in conjunction with IPRange.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FromPort

The start of the port range for port based protocols. For ICMP this specifies the type number.

Use -1 to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ToPort

The end of the port range for port based protocols. For ICMP this specifies the type number, where -1 can be used to specify all ICMP types.

Type:	Decimal
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The LiteralPath can be piped in.

Outputs

None

By default, this cmdlet returns no output.

Notes

Security groups cannot be removed in AWS if they are referened by rules from other security groups.

Security groups can be added and removed using the New-Item and Remove-Item cmdlets.

Related Links

- Amazon AuthorizeSecurityGroupEgress
- IANA protocol numbers
- Grant-HypSecurityGroupIngress
- Grant-HypSecurityGroupEgress
- Revoke-HypSecurityGroupIngress

Set-HypAdminConnection

March 11, 2024

Set the controller to be used by the cmdlets that form the Host service PowerShell snap-in.

Syntax

```
Set-HypAdminConnection
   [-AdminAddress <String>]
   [-BearerToken <String>]
   [-AdminClientIP <String>]
   [-TraceParent <String>]
   [-TraceState <String>]
   [<CommonParameters>]
```

Description

Use this command to set the default controller address to be used by the cmdlets to communicate with the controller. Most Host service cmdlets take an 'AdminAddress'parameter that can be used to define the controller (when used, this overrides this setting). However, the Set-Location cmdlet in the Host service provider does not offer this option. Therefore, the controller address must be set using this cmdlet, if no other cmdlets have set the address previously in the current runspace.

Examples

EXAMPLE 1

This command sets the controller address for the commands to be “myserver.com”. All commands run use this address until it is altered, either by another call to this command or by the use of the ‘AdminAddress’parameter in another command in the Host service snap-in.

```
1  Set-HypAdminConnection -AdminAddress myserver.com
```

Parameters

-AdminAddress

Specifies the address of a XenDesktop controller to which the PowerShell snap-in connects. You can provide this as a host name or an IP address.

Type:	String
Position:	Named
Default value:	LocalHost. Once a value is provided by any cmdlet, this value becomes the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-BearerToken

Specifies the bearer token assigned to the calling user

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminClientIP

Specifies the client IP of the calling user

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TraceParent

Specifies the trace parent assigned for internal diagnostic tracing use

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TraceState

Specifies the trace state assigned for internal diagnostic tracing use

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_HypHostSnapin

Set-HypDBConnection

March 11, 2024

Configures a database connection for the Host Service.

Syntax

```
Set-HypDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Specifies the database connection string for use by the currently selected Citrix Host Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1  Set-HypDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
       Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
1  Set-HypDBConnection -DBConnection $null
```


Parameters

-DBConnection

Specifies the database connection string to be used by the Host Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-HypDBConnection cmdlet returns an object describing the status of the Host Service together with extra diagnostics information. Possible values are:

- OK:

The Host Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Host Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Host Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Host Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Host Service currently in use is newer than, and incompatible with, the version of the Host Service schema on the database. Upgrade the Host Service to a more recent version.

- DBOlderVersionThanService:

The version of the Host Service schema on the database is newer than, and incompatible with, the version of the Host Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Host Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Host Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Host Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypServiceStatus
- Get-HypDBConnection
- Test-HypDBConnection

Set-HypDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Host Service.

Syntax

```
Set-HypDBCredentials
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-HypDBCredentials
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Host Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Host Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-HypDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.


```
$sqlCred = Get-Credential
Set-HypDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-HypDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-HypDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-HypDBSchema
- Set-HypDBConnection
- Get-Credential

Set-HypHostingUnitMetadata

March 11, 2024

Adds or updates metadata on the given HostingUnit.

Syntax

```
Set-HypHostingUnitMetadata
   [-HostingUnitUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHostingUnitMetadata
   [-HostingUnitUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHostingUnitMetadata
   [-HostingUnitName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHostingUnitMetadata
   [-HostingUnitName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHostingUnitMetadata
   [-InputObject] <HostingUnit[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHostingUnitMetadata
   [-InputObject] <HostingUnit[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given HostingUnit objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the HostingUnit with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-HypHostingUnitMetadata -HostingUnitUid 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                        Value
---                                        -----
property                                   value
```

Parameters

-HostingUnitUid

Id of the HostingUnit

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HostingUnitName

Name of the HostingUnit

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HostingUnit[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the HostingUnit specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-HypHostingUnitMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHostingUnitMetadata
- Remove-HypHostingUnitMetadata

Set-HypHostingUnitStorage

March 11, 2024

Sets options for a storage location on a hosting unit.

Syntax


```
Set-HypHostingUnitStorage
   [-LiteralPath] <String>
   [-StoragePath] <String>
   [-StorageType <StorageType>]
   [-Superseded <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this command to set options for storage locations that are defined for a hosting unit. Do not use this command if the connection for the hosting unit is in maintenance mode.

Examples

EXAMPLE 1

The command updates a storage location called “newStorage.storage”associated with the hosting unit called “MyHostingUnit”.

```
Set-HypHostingUnitStorage -LiteralPath XDHyp:\HostingUnits\
   MyHostingUnit -StoragePath 'XDHyp:\HostingUnits\MyHostingUnits\
   newStorage.storage' -Superseded $true

          HostingUnitUid       : bcd3d649-86d1-4aa8-8ec2-d322b6a2c457
          HostingUnitName      : MyHostingUnit
          HypervisorConnection : MyConnection
          RootPath             : /
          RootId               :
          NetworkPath          : /Network 0.network
          NetworkId            : ab47080b-ca15-771a-c8dc-6ad9650158f1
          Storage              : {
 /Local storage.storage, /newStorage.storage }

          PersonalvDiskStorage : {
 /newStorage.storage }

          VMTaggingEnabled     : True
          AdditionalStorage    : {
 TemporaryStorage }

          Metadata             : {
  }
```


Parameters

-LiteralPath

Specifies the path to the hosting unit which defines the storage. The path must be in one of the following formats:

<drive>:\HostingUnits\<HostingUnitName>

or <drive>:\HostingUnits{<HostingUnit Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StoragePath

Specifies the path to the storage that will be modified. The path must be in one of the following formats:

<drive>:\Connections\<ConnectionName>\MyStorage.storage

or <drive>:\Connections{<Connection Uid>}\MyStorage.storage

or <drive>:\HostingUnits\<HostingUnitName>\MyStorage.storage

or <drive>:\HostingUnits{<hostingUnit Uid>}\MyStorage.storage

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-StorageType

Specifies the type of storage in StoragePath. Supported storage types are:

OSStorage

PersonalvDiskStorage

TemporaryStorage

Type:	StorageType
Position:	Named
Default value:	OSStorage
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Superseded

Specifies that this storage has been superseded and must not be used for future provisioning operations. Existing virtual machines using this storage will continue to function.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path to Add-HypHostingUnitStorage (StoragePath parameter).

Outputs

Citrix.XDPowerShell.HostingUnit

Set-HypHostingUnitStorage returns an object containing the new definition of the hosting unit.

HostingUnitUid <Guid>

Specifies the unique identifier for the hosting unit.

HostingUnitName <string>

Specifies the name of the hosting unit.

HypervisorConnection <Citrix.XDPowerShell.HypervisorConnection>

Specifies the connection that the hosting unit uses to access a hypervisor.

RootId <string>

Identifies, to the hypervisor, the root of the hosting unit.

RootPath <string>

The hosting unit provider path that represents the root of the hosting unit.

Storage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use.

PersonalvDiskStorage <Citrix.XDPowerShell.Storage[]>

The list of storage items that the hosting unit can use for storing personal data.

VMTaggingEnabled <Boolean>

Specifies whether or not the metadata in the hypervisor can be used to store information about the XenDesktop Machine Creation Service.

NetworkId <string>

The hypervisor's internal identifier that represents the network specified for the hosting unit.

NetworkPath <string>

The hosting unit provider path to the network specified for the hosting unit.

AdditionalStorage

The list of additional storage items that the hosting unit can use.

Metadata <Citrix.XDPowerShell.Metadata[]>

A list of key value pairs that can store additional information about the hosting unit.

Notes

The storage path must be valid for the hosting unit. The rules that are applied are as follows:

XenServer (HypervisorConnection Type = XenServer)

NA

VMWare vSphere/ESX (HypervisorConnection Type = vCenter)

The storage path must be directly contained in the root path item of the hosting unit.

Microsoft SCVMM/Hyper-v (HypervisorConnection Type = SCVMM)

Only one storage entry for these connection types is valid, and it must reference an SMB share. Additionally, if a Hyper-V failover cluster is used the SMB share must be the top-level mount point of the cluster shared volume on one of the servers in the cluster (i.e. C:\ClusterStorage).

In the case of failure, the following errors can be produced.

Error Codes

HostingUnitsPathInvalid

The path provided is not to an item in a subdirectory of a hosting unit item.

HostingUnitStoragePathInvalid

The specified path is invalid.

HostingUnitStoragePathInvalid

The storage path cannot be found or is invalid. See notes above about validity.

HostingUnitStorageDuplicateObjectExists

The specified storage path is already part of the hosting unit.

HypervisorInMaintenanceMode

The hypervisor for the connection is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item
- Add-HypMetadata
- Remove-HypHostingUnitStorage

Set-HypHypervisorConnectionMetadata

March 11, 2024

Adds or updates metadata on the given HypervisorConnection.

Syntax

```
Set-HypHypervisorConnectionMetadata
   [-HypervisorConnectionUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHypervisorConnectionMetadata
   [-HypervisorConnectionUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHypervisorConnectionMetadata
   [-HypervisorConnectionName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHypervisorConnectionMetadata
   [-HypervisorConnectionName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-HypHypervisorConnectionMetadata
   [-InputObject] <HypervisorConnection[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypHypervisorConnectionMetadata
   [-InputObject] <HypervisorConnection[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given HypervisorConnection objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the HypervisorConnection with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-HypHypervisorConnectionMetadata -HypervisorConnectionUid 4CECC26E
   -48E1-423F-A1F0-2A06DDD0805C -Name property -Value value

Key                                      Value
---                                      -----
property                                 value
```

Parameters

-HypervisorConnectionUid

Id of the HypervisorConnection

Type:	Guid
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-HypervisorConnectionName

Name of the HypervisorConnection

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	HypervisorConnection[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the HypervisorConnection specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-HypHypervisorConnectionMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Add-HypHypervisorConnectionMetadata
- Remove-HypHypervisorConnectionMetadata

Set-HypServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-HypServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-HypServiceMetadata
   [-InputObject] <Service[]>
```



```
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-HypServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-HypServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Remove-HypServiceMetadata

Set-HypVolumeServiceConfiguration

March 11, 2024

Applies a change to one of the VolumeServiceConfiguration instances in the site.

Syntax

```
Set-HypVolumeServiceConfiguration
   -ConnectionType <String>
   -VolumeServiceConfigurationName <String>
   -VolumeWorkerPackageUri <String>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-HypVolumeServiceConfiguration
   -ConnectionType <String>
    -VolumeServiceConfigurationName <String>
    -RegionName <String>
    -BaseLinuxTemplateId <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Volume service configurations are used to control how cloud-based host connections behave when provisioning machines. They contain two pieces of information. The first is a per-region specification of the cloud template that provides the standard Linux operating system for the cloud. The second is the specification of a URL from which the Citrix Volume Worker software can be installed (not all cloud connections make use of this URL, but they all make use of the template map).

Each cmdlet invocation can be used to either change the volume worker URL, or to modify (or add) an entry in the Linux template map. These two operations are supported by parameter sets. To change both properties, you must invoke the cmdlet twice.

Examples

EXAMPLE 1

This command specifies a Linux template that should be used by default for CloudPlatform connections.

```
Set-HypVolumeServiceConfiguration -VolumeServiceConfigurationName
    SiteDefault -ConnectionType CloudPlatform -RegionName Region1 -
    BaseLinuxTemplateId 9883ba7a-74bf-4002-9b12-18bcc2c2fae8
```

EXAMPLE 2

This command specifies a worker package download URL that should be used by default for CloudPlatform connections.

```
Set-HypVolumeServiceConfiguration -VolumeServiceConfigurationName
    SiteDefault -ConnectionType CloudPlatform -VolumeWorkerPackageUri "
    http://cloudadmin.net/citrix/volumeworker/version1"
```


Parameters

-ConnectionType

Specifies the cloud connection type, such as “AWS”or “CloudPlatform”.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VolumeServiceConfigurationName

Specifies the name of the configuration you want to modify. This parameter is used alongside the ConnectionType to specify a single configuration set unambiguously. There can be only one named configuration per connection type. In a newly-configured site, there will be exactly one configuration set called “SiteDefault”for each of CloudPlatform and AWS.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VolumeWorkerPackageUri

Specifies a (new) URI for the volume worker package.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RegionName

Specifies the cloud region in which the Linux template resides. This parameter is used only when passing the BaseLinuxTemplateId parameter. The format of the string is cloud-specific. For example, for an AWS-based cloud, it would be a region identifier such as “us-east-1”.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-BaseLinuxTemplateId

Specifies a change to the standard Linux template that should be used in the cloud. When passing this parameter, you must also specify the RegionName parameter to indicate the region in which this template resides. The format of the string is cloud-specific. For example, for an AWS-based cloud, it would be an AMI identifier such as “ami-1234abcd”.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Host.Sdk.VolumeServiceConfiguration

This cmdlet returns the updated VolumeServiceConfiguration object.

Related Links

- Get-HypVolumeServiceConfiguration

Start-HypVM

March 11, 2024

Starts a VM.

Syntax

```
Start-HypVM
     [-LiteralPath] <String>
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

This command provides a mechanism to start a VM.

Examples

EXAMPLE 1

This command starts a VM called ‘MyVm.vm’within a hypervisor connection called ‘MyConnection’ .

```
Start-HypVM -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm
```

Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the virtual machine item to start. The path specified must be in one of the following formats: <drive>:\Connections\<Connection Name>\<Item Path of VM object> or <drive>:\Connections{<connection Uid>\<Item Path of VM object>} or

<drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>:\HostingUnits{<hostingUnit Uid>\<Item Path of VM object>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

InputHypervisorItemPathInvalid

The path provided is not to an item in a sub-directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

HypervisorInMaintenanceMode

The hypervisor is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin

Stop-HypVM

March 11, 2024

Stops a VM by issuing a Shutdown request

Syntax

```
Stop-HypVM
    [-LiteralPath] <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this command to change the power state of a VM from running to stopped.

Examples

EXAMPLE 1

This command stops a VM called 'MyVm.vm' within a hypervisor connection called 'MyConnection'.

```
Stop-HypVM -LiteralPath XDHyp:\Connections\MyConnection\MyVm.vm
```

Parameters

-LiteralPath

Specifies the path within a hosting unit provider to the virtual machine item to stop. The path specified must be in one of the following formats: <drive>:\Connections\<Connection Name>\<Item Path of VM object> or <drive>:\Connections{<connection Uid>\<Item Path of VM object>} or <drive>:\HostingUnits\<HostingUnit Name>\<Item Path of VM object> or <drive>:\HostingUnits{<hostingUnit Uid>\<Item Path of VM object>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a string that contains a path.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

InputHypervisorItemPathInvalid

The path provided is not to an item in a sub-directory of a connection item or a hosting unit item.

InvalidHypervisorItemPath

No item exists with the specified path.

InvalidHypervisorItem

The item specified by the path exists, but is not a VM Item.

HypervisorInMaintenanceMode

The hypervisor is in maintenance mode.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin

Test-HypDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Host Service.

Syntax

```
Test-HypDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-HypDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-HypDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Host Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Host SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-HypDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
   Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Host Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-HypDBConnection cmdlet returns an object describing the status of the selected Host Service instance that would result if the connection string were used with the Set-HypDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-HypDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Host Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Host Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Host Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Host Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Host Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Host Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_HypHostSnapin
- Get-HypServiceStatus
- Get-HypDBConnection
- Set-HypDBConnection

Test-HypHostingUnitNameAvailable

March 11, 2024

Checks to ensure that the proposed name for a hosting unit is unused.

Syntax

```
Test-HypHostingUnitNameAvailable
    -HostingUnitName <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this command to check that the proposed name for a hosting unit is unused. This check is done without regard for scoping of the existing hosting unit, so the names of inaccessible hosting units are also checked.

Examples

EXAMPLE 1

This tests whether the value of $NewHostingUnitName is unique or not, and can be used to create a new hosting unit or rename an existing one without failing. True is returned if the name is unique.


```
Test-HypHostingUnitNameAvailable -HostingUnitName $NewHostingUnitName
```

Parameters

-HostingUnitName

The name or names of the hosting units(s) to be tested.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

object[]

An array of PSObjects which pair the name and availability of the name

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item
- Rename-Item

Test-HypHypervisorConnectionNameAvailable

March 11, 2024

Checks to ensure that the proposed name for a hypervisor connection is unused.

Syntax

```
Test-HypHypervisorConnectionNameAvailable
    -HypervisorConnectionName <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Use this command to check that the proposed name for a hypervisor connection is unused. This check is done without regard for scoping of the existing hypervisor connection, so the names of inaccessible hypervisor connection are also checked.

Examples

EXAMPLE 1

This tests whether the value of $NewConnectionName is unique or not, and can be used to create a new hypervisor connection or rename an existing one. True is returned if the name is unique.

```
Test-HypHypervisorConnectionNameAvailable -HypervisorConnectionName
    $NewConnectionName
```

Parameters

-HypervisorConnectionName

The name or names of the hypervisor connection(s) to be tested.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

object[]

An array of PSObjects which pair the name and availability of the name

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin
- New-Item
- Rename-Item

Test-HypPvsCollectionNameAvailable

March 11, 2024

Tests availability of collection name.

Syntax

```
Test-HypPvsCollectionNameAvailable
    -SiteId <Guid>
    -CollectionName <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Test the availability of a collection name against a site.

Examples

EXAMPLE 1

This command tests the availability of a Collection Name.

```
Test-HypPvsCollectionNameAvailable -SiteId
    00000000-0000-0000-0000-000000000000 -CollectionName "CollectionName
    "
```

Parameters

-SiteId

Target Site ID to validate against.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-CollectionName

Collection name to test.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Guid

The GUID of a Site to validate against.

String

Collection name to test availability for.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Register-HypPvsSite
- Unregister-HypPvsSite

Unregister-HypPvsServer

March 11, 2024

Unregister PVS Server.

Syntax

```
Unregister-HypPvsServer
           -ServerId <Guid>
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```

Description

Use this command to remove a PvsServer with the corresponding ServerId

Examples

EXAMPLE 1

This command unregisters the PVS server with ServerId 00000000-0000-0000-0000-000000000000

```
Unregister-HypPvsServer -ServerId 00000000-0000-0000-0000-000000000000
```

Parameters

-ServerId

The ID of the PVS Server.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- Register-HypPvsServer

Unregister-HypPvsSite

March 11, 2024

Unregister PVS Site.

Syntax

```
Unregister-HypPvsSite
           -SiteId <Guid>
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```

Description

Use this command to remove a PvsSite with the corresponding SiteId

Examples

EXAMPLE 1

This command unregisters the PVS site with SiteId 00000000-0000-0000-0000-000000000000

```
Unregister-HypPvsSite -SiteId 00000000-0000-0000-0000-000000000000
```


Parameters

-SiteId

The ID of the PVS site.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- Get-HypPvsSite
- Register-HypPvsSite

Update-HypHypervisorConnection

March 11, 2024

Requests the host service to update the connection properties that depend on the version

of hypervisor in use.

Syntax

```
Update-HypHypervisorConnection
      [-LoggingId <Guid>]
      [-LiteralPath] <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Use this command to update the version-specific properties of a hypervisor connection, after an upgrade to the hypervisor system which may provide new capabilities.

Examples

EXAMPLE 1

This command requests that the properties of Connection1 be updated to match the current capabilities of the hypervisor.

```
Update-HypHypervisorConnection -LiteralPath xdhyp:\connections\
   Connection1
```

Parameters

-LiteralPath

Specifies the path within a Host Service provider to the hypervisor connection item to be updated. The path specified must be in one of the following formats; <drive>:\Connections\<HypervisorConnectionName> or <drive>:\Connections{HypervisorConnection Uid>}

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

InvalidPath

The path provided is not in the required format.

HypervisorInMaintenanceMode

The hypervisor is in maintenance mode and cannot be updated.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for

various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_HypHostSnapin

about_ProvMachineCreationSnapIn

March 11, 2024

Topic

about_ProvMachineCreationSnapin

Short Description

The Machine Creation Service PowerShell snap-in provides administrative functions for the Machine Creation Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Prov'.

Long Description

The Machine Creation Service PowerShell snap-in enables both local and remote administration of the Machine Creation Service. It provides facilities to create virtual machines and manage the associated disk images.

The snap-in provides two main entities:

Provisioning Scheme

Specifies details of new virtual machines created by the Machine Creation Service. Provisioning schemes define the following information.

Hosting Unit

Provides details of the hypervisor and storage on which new virtual machines will be created. Stored and maintained by the Host Service and PowerShell snap-in.

Identity Pool

Lists the Active Directory computer accounts available for use by new virtual machines. Stored and maintained by the Active Directory Identity Service and PowerShell snap-in.

Master Image

Specifies the disk image that will be used for new virtual machines. Accessed through the hosting provider in the Host Service snap-in.

Provisioned VM

Defines the virtual machines created by the Machine Creation Services. These virtual machines are associated with the provisioning scheme from which they were created.

The processes of creating provisioning schemes and new virtual machines can take a significant amount of time to complete. For this reason, these long-running tasks can be run asynchronously so that other commands are accessible while the processes are running. Note, however, that only one long-running task can operate on a provisioning scheme at any one time. The processes are monitored using the Get-ProvTask command. For more information, see the help for Get-ProvTask.

about_ProvMachineCreationSnapIn

March 11, 2024

Topic

about_ProvMachineCreationSnapin

Short Description

The Machine Creation Service PowerShell snap-in provides administrative functions for the Machine Creation Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Prov'.

Long Description

The Machine Creation Service PowerShell snap-in enables both local and remote administration of the Machine Creation Service. It provides facilities to create virtual machines and manage the associated disk images.

The snap-in provides two main entities:

Provisioning Scheme

Specifies details of new virtual machines created by the Machine Creation Service. Provisioning schemes define the following information.

Hosting Unit

Provides details of the hypervisor and storage on which new virtual machines will be created. Stored and maintained by the Host Service and PowerShell snap-in.

Identity Pool

Lists the Active Directory computer accounts available for use by new virtual machines. Stored and maintained by the Active Directory Identity Service and PowerShell snap-in.

Master Image

Specifies the disk image that will be used for new virtual machines. Accessed through the hosting provider in the Host Service snap-in.

Provisioned VM

Defines the virtual machines created by the Machine Creation Services. These virtual machines are associated with the provisioning scheme from which they were created.

The processes of creating provisioning schemes and new virtual machines can take a significant amount of time to complete. For this reason, these long-running tasks can be run asynchronously so that other commands are accessible while the processes are running. Note, however, that only one long-running task can operate on a provisioning scheme at any one time. The processes are monitored using the Get-ProvTask command. For more information, see the help for Get-ProvTask.

about_Prov_CustomProperties

March 28, 2024

Topic

Citrix Machine Creation Service SDK - Custom Properties

Short Description

Overview of custom properties settings of the provisioning scheme.

Long Description

Custom properties are the properties of the provisioning scheme that are specific to the target hosting infrastructure. Custom properties can be used to control or tune behaviors of the provisioning scheme.

Custom Properties For Azure

The following custom properties are specific to hosting infrastructure targets using the AzureRM plugin.

- StorageAccountType: Deprecated. This property will be removed in the future, use property StorageType instead.
- StorageType: Storage type used for provisioned virtual machine disks. Storage types include: Standard_LRS, StandardSSD_ZRS, Premium_ZRS, StandardSSD_LRS and Premium_LRS. If this property is not specified, Standard_LRS is used by default.
- IdentityDiskStorageType: Storage type used for the provisioned virtual machine identity disk. If this property is not specified, StandardSSD_LRS is used by default.
- WBCDiskStorageType: Storage type for the provisioned virtual machine write back cache disk. This property only applies to a provisioning scheme with UseWriteBackCache enabled. If this property is not specified, the value specified in the StorageType property is used for WBCDiskStorageType.

- StorageTypeAtShutdown: Storage type for the provisioned virtual machine disks when it's powered down. It applies to OS disk and write back cache disk only if the provisioning scheme has UseWriteBackCache enabled. When the machine is powered back on, its storage type will be switched back to StorageType property for OS disk and WBCDiskStorageType for write back cache disk. If WBCDiskStorageType is not specified, it will default to StorageType. Default value for this property is empty. Standard_LRS will result in storage type change at shutdown. Any other value will result in an error. Also, when StorageType is selected as either StandardSSD_ZRS or Premium_ZRS, setting StorageTypeAtShutdown to Standard_LRS will result in an error too as this feature is not compatible with ZRS.
- ResourceGroups: Resource groups used to contain provisioned virtual machines. If this property is not specified, Citrix managed resource groups are created. Once the provisioning scheme is created, this property cannot be changed from Citrix managed resource groups to user created resource groups, and vice versa. It can be changed only from one user created resource groups set to another.
- LicenseType: The license type used for provisioned virtual machines. The valid values include Windows_Server, Windows_Client, RHEL_BYOS, and SLES_BYOS. Windows_Server and Windows_Client are for Windows OS. RHEL_BYOS and SLES_BYOS are for Linux OS. If LicenseType is not provided or a null value, Azure will charge the license.
- OsType: Operating system type used for the provisioned virtual machine. OS types include either Windows or Linux. If an OS type is not specified, Windows is used by default.
- UseManagedDisks: Indicate whether to use Azure managed disks for the provisioned virtual machine. Specify either True or False. If this property is not specified, the default value is determined upon the SchemaVersion custom property value (See SchemaVersion). Once the provisioning scheme is created, this property cannot be changed.
- PersistWBC: Persist the write back cache disk for the non-persistent provisioned virtual machine between power cycles. Specify either True or False. This property only applies to a provisioning scheme with UseWriteBackCache enabled. If this property is not specified, the write back cache disk is deleted when the virtual machine is shut down, and is re-created when the virtual machine is powered on.
- PersistOsDisk: Persist the OS disk when power cycling the non-persistent

provisioned virtual machine. Specify either True or False. This property only applies to a provisioning scheme with UseWriteBackCache enabled. If this property is not specified, the OS disk is deleted when the virtual machine is shut down, and is re-created when the virtual machine is powered on.

- PersistVm: Persist the non-persistent provisioned virtual machines in Azure environments when power cycling. Specify either True or False. This property only applies when the PersistOsDisk property is set to True. If this property is not specified, they will be deleted from Azure when powered off.
- MachinesPerStorageAccount: The maximum number of virtual machines containing unmanged disks that can be provisioned under a single Azure storage account. This property only applies when the UseManagedDisks property is set to False. If this property is not specified, a maximum value of 40 is used by default.
- StorageAccountsPerResourceGroup: The maximum number of Azure storage accounts provisioned under a single Azure resource group. This property only applies when the UseManagedDisks property is set to False. If this property is not specified, a maximum value of 100000 is used by default.
- DiskEncryptionSetId: The Azure Encryption Set ID that references the customer-managed encryption used for the server-side encryption key to encrypt disks for provisioned virtual machines. The supplied value must be a valid Azure resource ID. This property only applies when the UseManagedDisks property is set to True. If this property is not specified, Citrix takes no action on the provisioned virtual machines. Virtual machine disks use Azure storage encryption for data at rest. Once the provisioning scheme is created, this property cannot be changed.
- UseSharedImageGallery: Use Azure Shared Image Gallery as a published image repository for MCS to provision virtual machines in Azure. Specify either True or False. This property only applies when the UseManagedDisks property is set to True. If this property is not specified, UseSharedImageGallery is set to False by default, and the prepared images for the machine catalog are stored as managed disk snapshots.
- SharedImageGalleryReplicaRatio: The ratio of virtual machines to shared image gallery version replicas. This value is an integer greater than 0. This property only applies when the UseSharedImageGallery property is set to True. If this property is not specified,

SharedImageGalleryReplicaRatio is set to 1000 for persistent OS disks and 40 for non-persistent OS disks by default.

- SharedImageGalleryReplicaMaximum: The maximum number of replicas for each shared image gallery version. This value is an integer greater than 0. This property only applies when the UseSharedImageGallery property is set to True. If this property is not specified, SharedImageGalleryReplicaMaximum is set to 10 by default.
- DedicatedHostGroupId: The Azure Dedicated Host Group ID. Format the value as "myResourceGroup/myHostGroup". If this property is used, virtual machines are provisioned on the Azure dedicated hosts specified by DedicatedHostGroupId.
- EnableIntuneEnroll: Automatically enroll the provisioned virtual machines with Microsoft Intune to manage them with policy and apps at scale. This property only applies to a provisioning scheme with IdentityType set to "AzureAd". If this property is not specified, the provisioned virtual machine will not be enrolled with Intune.
- Zones: The Azure Availability Zones containing provisioned virtual machines. Use a comma as a delimiter for multiple zones. If this property is not specified, the provisioned virtual machine is randomly placed across all Azure Availability Zones in the region defined by the hosting unit.
- UseEphemeralOsDisk: Use Azure Ephemeral OS disk for the provisioned virtual machine. Specify either True or False. This property only applies when the UseManagedDisks property is not False and the UseSharedImageGallery property is set to True. When the UseEphemeralOsDisk property is set to True, the chosen virtual machine size supports the size of the Ephemeral OS disk and the cache disk, which is larger than the master image disk. If this property is not specified, UseEphemeralOsDisk is set to False by default. Once the provisioning scheme is created, this property cannot be changed.
- SchemaVersion: This value is a float and is added (value as float 2) automatically to CustomProperties if omitted in CustomProperties. If this property is set to greater than or equal to 2, MCS will use managed disks by default, unless UseManagedDisks property is set to False. If this property is set to less than 2, MCS will use unmanaged disks by default, unless UseManagedDisks property is set to True.

- UseTempDiskForWBC: Indicates whether to use Azure temporary storage to store write back cache file. Specify either True or False. If this property is not specified, the UseTempDiskForWBC parameter is set to False by default.
  The Azure temporary storage can be used to store write back cache file when all the following conditions are meet:

  PersistWBC is False
  UseTempDiskForWBC is True
  UseWriteBackCache is True
  UseEphemeralOsDisk is False
  Azure temporary storage has enough space for both paging file and write back cache file.

- PageFileDiskDriveLetterOverride: The drive letter of the disk where the page file is stored. The drive letter should be between ‘C’and ‘Z’.
  The properties ‘PageFileDiskDriveLetterOverride’, ‘InitialPageFileSizeInMB’, and ‘MaxPageFileSizeInMB’should always come together.

- InitialPageFileSizeInMB: The initial page file size in megabytes which must be between 16 MB and 16777216 MB, and not greater than the maximum page file size.

- MaxPageFileSizeInMB: The maximum page file size in megabytes which must be greater than or equal to the initial page file size, and less than 16777216 MB. Also it can not exceed the amount of free space on the drive.

- BackupVmConfiguration: A secondary list of service offerings optionally paired with type as Spot
  to fall back on when machine power on fails due to resource shortage error from Azure.
  Type when not specified will be inferred as Regular.
  We would always try to power on with the primary service offering first, and only when that fails we
  would try secondary config starting from the first item in the list.
  Failure to power on with primary or any of the secondary configs will result in a “Resource Not Available”error.
  Few examples of how to specify this property:
  “<CustomProperties>
  <Property xsi:type=“‘StringProperty‘”Name=“‘BackupVmConfiguration‘”Value=‘”[‘ServiceOffering’: ‘A’, ‘ServiceOffering’: ‘B’, ‘ServiceOffering’: ‘C’]‘”/></CustomProperties>”
  <Property xsi:type=“‘StringProperty‘”Name=“‘BackupVmConfiguration‘”Value=‘”[{‘ServiceOffering’: ‘A’, ‘Type’: ‘Spot’}, {‘ServiceOffering’: ‘B’, ‘Type’: ‘Regular’}, {‘ServiceOffering’: ‘C’, ‘Type’: ‘Spot’}]‘”/></CustomProperties>”

Custom Properties For Aws

The following custom properties are specific to hosting infrastructure targets using the AWS plugin.

- AwsCaptureInstanceProperties: Capture AWS instance properties, including the Identity Access Management role and tags from the master image, then apply them to the provisioned virtual machines. Specify either True or False. If this property is not specified, AwsCaptureInstanceProperties is set to False by default.
  Note: AwsCaptureInstanceProperties is now deprecated.
- AwsOperationalResourcesTagging: Apply tags from master image to all Citrix created operational resources including virtual machines, VM disks, VM network interfaces, S3 buckets, S3 objects, launch templates and AMIs. Specify either True or False. When setting this property to True, you must also set the AwsCaptureInstanceProperties parameter to True or must have supplied Machine Profile. If neither of these two values are specified, the AwsOperationalResourcesTagging parameter is set to False by default.

Custom Properties For Gcp

The following custom properties are specific to hosting infrastructure targets using the GCP plugin.

- CatalogZones: GCP zones to place provisioned virtual machines. Use comma as delimiter for multiple zones. If this property is not specified, the provisioned virtual machine is randomly placed across GCP zones.
- CryptoKeyId: Use the customer-managed encryption key to encrypt data on provisioned virtual machines. Format the value as "project:location:keyRing:key".If this property is not specified, a google managed encryption key is used.
- StorageType: Storage type used for provisioned virtual machine disks. Storage types depend on the region/zone, but can include: pd-standard, pd-balanced, pd-ssd, local-ssd, and pd-extreme. If this property is not specified, the master image's boot disk type is used by default.
- IdentityDiskStorageType: Storage type used for the provisioned virtual machine identity disk. If this property is not specified, 'pd-standard' is used by default.
- WBCDiskStorageType: Storage type for the provisioned virtual machine write back cache disk. This property only applies to a provisioning

scheme with UseWriteBackCache enabled. If this property is not specified, ‘pd-standard’is used by default.
- PersistOsDisk: Persist the OS disk when power cycling the non-persistent provisioned virtual machine. Specify either True or False. This property only applies to a provisioning scheme with UseWriteBackCache enabled. If this property is not specified, the OS disk is deleted when the virtual machine is shut down, and is re-created when the virtual machine is powered on.
- PersistWBC: Persist the write back cache disk for the non-persistent provisioned virtual machine between power cycles. Specify either True or False. This property only applies to a provisioning scheme with UseWriteBackCache enabled. If this property is not specified, the write back cache disk is deleted when the virtual machine is shut down, and is re-created when the virtual machine is powered on.

Custom Properties For Vmware

The following custom properties are specific to hosting infrastructure targets using the VMware plugin.

- FolderId: The ID of a VM Placement Folder. If a valid value for the FolderID is specified, for example, “group-v1234”, VMs are created in that folder ID. If the property is not specified, VMs are created in the same folder where the master image is placed.

Custom Properties For Scvmm

The following custom properties are specific to hosting infrastructure targets using the SCVMM plugin.

- AzureArcSubscriptionId: Azure subscription id that the virtual machine will be connected to as an Azure Arc connected virtual machine. If this property is not specified, the virtual machine will fail to connect to Azure Arc.
- AzureArcRegion: Azure region that the virtual machine will be connected to as an Azure Arc connected virtual machine. If this property is not specified, the virtual machine will fail to connect to Azure Arc.
- AzureArcResourceGroup: Azure resource group that the virtual machine will be connected to as an Azure Arc connected virtual machine. If this property is not specified, the virtual machine will fail to connect to Azure Arc.

- EnableAzureArcOnboarding: Indicates whether to connect the MCS provisioned virtual machines to Azure Arc. Specify either True or False.
  This property only applies to the SCVMM managed Azure Stack HCI cluster.

See Also

- New-ProvScheme
- Set-ProvScheme

about_Prov_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a
set of property names separated by commas, semi-colons, or spaces.
Optionally, prefix each name with a + or - to indicate ascending or
descending order, respectively. Ascending order is assumed if no
prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are
applied. For example, to sort by Name and then by Count (largest first)
use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value
of the elements. You can specify a different sort order by qualifying
the name with an ordered list of elements or their numeric values,
or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and
supports combination of conditions with -and and -or, and grouping
with braces. For example:

Get-<Noun> -Filter 'Name -like ''High*''-or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use
script blocks in most cases. This can be easier to read as it reduces
quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double

quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago
```

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned

property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID

that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Add-ProvImageVersionMetadata

March 11, 2024

Adds metadata on the given image version scheme.

Syntax

```
Add-ProvImageVersionMetadata
   [-PreparedImageVersionUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-ProvImageVersionMetadata
    [-PreparedImageVersionUid] <Guid>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-ProvImageVersionMetadata
    [-PreparedImageDefinitionName] <String>
    [-PreparedImageVersionNumber] <String>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-ProvImageVersionMetadata
    [-PreparedImageDefinitionName] <String>
    [-PreparedImageVersionNumber] <String>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-ProvImageVersionMetadata
    [-InputObject] <ImageVersion[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Add-ProvImageVersionMetadata
    [-InputObject] <ImageVersion[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given image version objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ImageVersionMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the image version with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Add-ProvImageVersionMetadata -PreparedImageVersionUid 4CECC26E-48E1-423
    F-A1F0-2A06DDD0805C -Name property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-PreparedImageVersionUid

Id of the image verison

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-PreparedImageDefinitionName

Name of the image definition

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-PreparedImageVersionNumber

Version number of the image version

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

ImageVersion objects to which the metadata is to be added.

Type:	ImageVersion[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the image version specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageVersion

You can pipe a ImageVersion object or any object containing a parameter called 'ImageDefinitionName'and 'ImageVersionNumber'or 'ImageVersionUid'to Add-ImageVersionMetadata.

PSObject

A metadata map object can be piped to the Add-ImageVersionMetadata command.

Outputs

Citrix.MachineCreation.Sdk.Metadata

Add-ProvImageVersionMetadata returns an array of objects containing the newly added metadata.

Property <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes ————InvalidParameterCombination

The cmdlet parameters are inconsistent.

ImageVersionNotFound

One of the specified image version was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvImageVersionMetadata
- Remove-ProvImageVersionMetadata

Add-ProvMetadataConfiguration

March 11, 2024

Add VM metadata configuration settings for a plugin.

Syntax

```
Add-ProvMetadataConfiguration
   [-PluginType] <String>
   [-ConfigurationName] <String>
   [-ConfigurationValue] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to configure metadata setting for a hypervisor. Customer can run the command to manage the allowed Azure VM extension list.

When an extension is added, it will be installed for catalog machines if the extension is also defined in MachineProfile VM/Template.

Extensions that are not in the allowed list will be automatically ignored during catalog creation.

Examples

EXAMPLE 1

Add Azure VM extension AADLoginForWindows

```
Add-ProvMetadataConfiguration -PluginType "AzureRmFactory" -
   ConfigurationName "Extension" -ConfigurationValue "
   AADLoginForWindows"
```


Parameters

-PluginType

The name of the hypervisor plug-in factory. Currently, AzureRmFactory is the only supported plug-in factory.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ConfigurationName

The configuration name. Currently, Extension is the only supported configuration.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ConfigurationValue

The setting for metadata configuration, for example Azure extension type.

Type:	String
Position:	4
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ConfigurationValueAlreadyExists

The configuration has already been added to the database.

ConfigurationNotValid

The plugin type or configuration name is not valid. AzureRmFactory is the only plugin type, and Extension is the only configuration supported.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, check the Windows event logs on your self-hosted delivery controller or contact Citrix support if using Citrix DaaS (Citrix-hosted delivery controller).

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvMetadataConfiguration
- Remove-ProvMetadataConfiguration

Add-ProvSchemeControllerAddress

March 11, 2024

Adds a list of controller addresses to a provisioning scheme, not recommended any more due to an enhancement in that the list of controller address can be automatically retrieved from the broker.

Syntax

```
Add-ProvSchemeControllerAddress
   [-ProvisioningSchemeName] <String>
   [-ControllerAddress] <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeControllerAddress
   -ProvisioningSchemeUid <Guid>
   [-ControllerAddress] <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to associate controller hosts, or a set of brokers, with a specific provisioning scheme. This optional data is passed to the machines created by the Machine Creation Services, where it is used to associate the newly created machine with a broker. The list is returned along with the assigned provisioning scheme.

Examples

EXAMPLE 1

Add a set of controllers to the provisioning scheme with the identifier "01a4a008-8ce8-4165-ba9c-cdf15a6b0501".

```
Add-ProvSchemeControllerAddress -ProvisioningSchemeUid "01a4a008-8ce8
   -4165-ba9c-cdf15a6b0501" -ControllerAddress (ddcA.citrix.com,ddcB.
   citrix.com,ddcC.citrix2.com)

CleanOnBoot                   : True
ControllerAddress             : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }

CpuCount                      : 1
DiskSize                      : 20
HostingUnitName               : HostUnit1
HostingUnitUid                : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName              : idPool1
IdentityPoolUid               : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                  : 0
MachineProfile                :
MasterImageVM                 : Base.vm/base.snapshot
MasterImageVMDate             : 17/05/2020 09:53:40
MemoryMB                      : 1024
Metadata                      : {
 Department = Sales }

MetadataMap                   : {
 [Department = Sales] }

ProvisioningSchemeName        : Scheme2
ProvisioningSchemeUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeVersion     : 1
TaskId                        : 00000000-0000-0000-0000-000000000000
VMMetadata                    : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter      :
PersonalVDiskDriveSize        : 0
UsePersonalVDiskStorage       : False
NetworkMaps                   : {
 0 }

Scopes                        :
DedicatedTenancy              : False
GpuTypeId                     :
ResetAdministratorPasswords   : False
SecurityGroups                : {
  }

```



```
ServiceOffering                :
TenancyType                    : Shared
AzureAdJoinType                :
CurrentMasterImageUid          : c0571690-4f57-4476-901b-fe64d6aecb79
CustomProperties               :
IdentityType:                  : ActiveDirectory
UseFullDiskCloneProvisioning   : False
UseWriteBackCache              : True
WriteBackCacheDiskSize         : 24
WriteBackCacheMemorySize       : 256
Warnings                       : {
  }

WriteBackCacheDiskIndex        : 0
```

EXAMPLE 2

Add controller addresses to a provisioning scheme using a ProvisioningScheme object.

```
Get-ProvScheme -ProvisioningSchemeName scheme1 | Add-
    ProvSchemeControllerAddress -ControllerAddress (ddcA.citrix.com,ddcB
    .citrix.com,ddcC.citrix2.com)

CleanOnBoot                    : True
ControllerAddress              : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }

CpuCount                       : 1
DiskSize                       : 20
HostingUnitName                : HostUnit1
HostingUnitUid                 : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName               : idPool1
IdentityPoolUid                : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                   : 0
MachineProfile                 :
MasterImageVM                  : Base.vm/base.snapshot
MasterImageVMDate              : 17/05/2020 09:53:40
MemoryMB                       : 1024
Metadata                       : {
 Department = Sales }

MetadataMap                    : {
 [Department = Sales] }

ProvisioningSchemeName         : Scheme2
ProvisioningSchemeUid          : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeVersion      : 1
TaskId                         : 00000000-0000-0000-0000-000000000000
VMMetadata                     : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter       :
```



```
PersonalVDiskDriveSize        : 0
UsePersonalVDiskStorage       : False
NetworkMaps                   : {
 0 }

Scopes                        :
DedicatedTenancy              : False
GpuTypeId                     :
ResetAdministratorPasswords   : False
SecurityGroups                : {
  }

ServiceOffering               :
TenancyType                   : Shared
AzureAdJoinType               :
CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
CustomProperties              :
IdentityType:                 : ActiveDirectory
UseFullDiskCloneProvisioning  : False
UseWriteBackCache             : True
WriteBackCacheDiskSize        : 24
WriteBackCacheMemorySize      : 256
Warnings                      : {
  }

WriteBackCacheDiskIndex       : 0
```

Parameters

-ProvisioningSchemeName

The name for the provisioning scheme that the list of addresses is to be added to.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme that the list of addresses is to be added to.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerAddress

Specifies the array of controller DNS names to be added to the provisioning scheme.

Type:	String[]
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Add-ProvSchemeControllerAddress.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

Add-ProvSchemeControllerAddress returns the updated ProvisioningScheme object containing the union of the old and new controller address lists.

CleanOnBoot <bool>

Indicates whether the VMs created will be reset to a clean state on each start.

ControllerAddress <string[]>

The DNS names of the controllers associated with this provisioning scheme.

CpuCount <int>

The number of processors that will be used to create VMs.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

HostingUnitName <string>

The name of the hosting unit being used by this provisioning scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this provisioning scheme.

IdentityPoolName <string>

The name of the identity pool being used by this provisioning scheme.

IdentityPoolUid <Guid>

The unique identifier of the identity pool being used by this provisioning scheme.

MachineCount <int>

The count of machines created with this provisioning scheme.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

MasterImageVM <string>

The inventory path to the VM snapshot copy used by this provisioning scheme.

MasterImageVMDate <DateTime>

The date and time when the VM snapshot copy used by this provisioning scheme was made.

MemoryMB <int>

The maximum amount of memory that will be used to created VMs in MB.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The metadata associated with this provisioning scheme.

MetadataMap <IDictionary[string, string];>

The metadata associated with this provisioning scheme arranged in key value pairs.

ProvisioningSchemeName <string>

The name of this provisioning scheme.

ProvisioningSchemeUid <Guid>

The unique identifier for this provisioning scheme.

ProvisioningSchemeVersion <int>

The version of the provisioning scheme.

TaskId <Guid>

The identifier of any current task that is running for the provisioning scheme.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

PersonalVDiskDriveLetter <char>

The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int>

The size of the personal vDisk in GB.

UsePersonalVDiskStorage <bool>

Indicates whether this provisioning scheme uses personal vDisk storage.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

Scope <Citrix.MachineCreation.Sdk.ScopeReference[]>

The administration scopes associated with this provisioning scheme.

DedicatedTenancy <bool>

Indicates whether dedicated tenancy is used when creating VMs in Cloud Hypervisors.

GpuTypeId <string>

The id of the GPU (Graphics Processor Unit) Type used by this scheme. It is null if there is no GPU.

ResetAdministratorPasswords <bool>

Indicates whether the passwords for administrator accounts are reset on created machines.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

TenancyType <string>

Tenancy type to be used when creating VMs in Cloud Hypervisors. (See New-ProvScheme.)

AzureAdJoinType <string>

Deprecated.

CurrentMasterImageUid <Guid>

The unique identifier of the current master image used by the provisioning scheme. (See Get-ProvSchemeMasterVMImageHistory.)

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

IdentityType <string>

Identity type used to join created machines to a directory service. (See New-ProvScheme.)

UseFullDiskCloneProvisioning <bool>

Indicates whether VMs will be created using the dedicated full disk clone feature.

UseWriteBackCache <bool>

Indicates whether this provisioning scheme will use the write-back cache feature.

WriteBackCacheDiskSize <int>

The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int>

The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvScheme
- Remove-ProvSchemeControllerAddress

Add-ProvSchemeMetadata

March 11, 2024

Adds metadata on the given provisioning scheme.

Syntax

```
Add-ProvSchemeMetadata
   [-ProvisioningSchemeUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Add-ProvSchemeMetadata
   [-ProvisioningSchemeUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeMetadata
   [-ProvisioningSchemeName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeMetadata
   [-ProvisioningSchemeName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeMetadata
   [-InputObject] <ProvisioningScheme[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeMetadata
   [-InputObject] <ProvisioningScheme[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given provisioning scheme objects. This cmdlet will not overwrite existing metadata on an object - use the Set-ProvSchemeMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the provisioning scheme with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Add-ProvSchemeMetadata -ProvisioningSchemeUid 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-ProvisioningSchemeUid

Id of the ProvisioningScheme

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeName

Name of the ProvisioningScheme

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

ProvisioningScheme objects to which the metadata is to be added.

Type:	ProvisioningScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the provisioning scheme specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe a ProvisioningScheme object or any object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Add-ProvSchemeMetadata.

PSObject

A metadata map object can be piped to the Add-ProvSchemeMetadata command.

Outputs

Citrix.MachineCreation.Sdk.Metadata

Add-ProvSchemeMetadata returns an array of objects containing the newly added metadata.

Property <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvSchemeMetadata
- Remove-ProvSchemeMetadata

Add-ProvSchemeScope

March 11, 2024

Add the specified provisioning scheme(s) to the given scope(s).

Syntax

```
Add-ProvSchemeScope
   [-Scope] <String[]>
   -InputObject <ProvisioningScheme[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeScope
   [-Scope] <String[]>
   -ProvisioningSchemeUid <Guid[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvSchemeScope
   [-Scope] <String[]>
   -ProvisioningSchemeName <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The Add-ProvSchemeScope command is used to associate one or more provisioning schemes with given scope(s).

To add a provisioning scheme to a scope, you need permission to change the scopes of the provisioning scheme and permission to add objects to all of the scopes you have specified.

If the provisioning scheme is already in a scope, that scope will be silently ignored.

Examples

EXAMPLE 1

Adds a single provisioning scheme to the 'Finance'scope.


```
Add-ProvSchemeScope Finance -ProvisioningSchemeUid 6702C5D0-C073-4080-
    A0EE-EC74CB537C52
```

EXAMPLE 2

Adds a single provisioning scheme to the multiple scopes.

```
Add-ProvSchemeScope Finance,Marketing -ProvisioningSchemeUid 6702C5D0-
    C073-4080-A0EE-EC74CB537C52
```

EXAMPLE 3

Adds all visible provisioning schemes to the 'Finance'scope.

```
Get-ProvScheme | Add-ProvSchemeScope Finance
```

EXAMPLE 4

Adds all provisioning schemes whose name starts with letter 'A'to the 'Finance'scope.

```
Add-ProvSchemeScope Finance -ProvisioningSchemeName A*
```

Parameters

-Scope

Specifies the scopes to add the objects to.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

Specifies the ProvisioningScheme objects to be added.

Type:	ProvisioningScheme[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

Specifies the ProvisioningScheme objects to be added by the UID of the provisioning scheme.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeName

Specifies the ProvisioningScheme objects to be added by the name of the provisioning scheme.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe a ProvisioningScheme object or any object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Add-ProvScope.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvSchemeScope
- Get-ProvScopedObject

Add-ProvTaskMetadata

March 11, 2024

Adds metadata on the given Task.

Syntax

```
Add-ProvTaskMetadata
   [-TaskId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvTaskMetadata
   [-TaskId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvTaskMetadata
   [-InputObject] <Task[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-ProvTaskMetadata
   [-InputObject] <Task[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Use this cmdlet to store additional custom data against given Task objects.

This cmdlet does not overwrite existing metadata on an object - use the Set-ProvTaskMetadata cmdlet instead.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Add-ProvTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name
    property -Value value

Property                                 Value
--------                                 -----
property                                 value
```

Parameters

-TaskId

Id of the Task.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	Task[]

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Add-ProvTaskMetadata command.

PSObject

A metadata map object can be piped to the Add-ProvTaskMetadata command.

Outputs

Citrix.MachineCreation.Sdk.Metadata

Add-ProvTaskMetadata returns an array of objects containing the newly added metadata.

Property <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can result.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DuplicateObject

One of the specified metadata already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvTaskMetadata
- Remove-ProvTaskMetadata
- Get-ProvTask
- Stop-ProvTask
- Remove-ProvTask
- Switch-ProvTask

Cancel-ProvVMUpdate

March 11, 2024

(DEPRECATED) Cancels pending property updates on provisioned virtual machines.

Syntax

```
Cancel-ProvVMUpdate
      -ProvisioningSchemeName <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Cancel-ProvVMUpdate
      -ProvisioningSchemeName <String>
      -VMName <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Cancel-ProvVMUpdate
      -ProvisioningSchemeUid <Guid>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Cancel-ProvVMUpdate
      -ProvisioningSchemeUid <Guid>
      -VMName <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet has been deprecated and replaced by %5BClear-ProvVMUpdateTimeWindow%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Clear-ProvVMUpdateTimeWindow.html). %5BClear-ProvVMUpdateTimeWindow%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Clear-ProvVMUpdateTimeWindow.html) is the new name for Cancel-ProvVMUpdate and provides all the same functionality, please use it going foward in place of Cancel-ProvVMUpdate.

This cmdlet clears the ProvisioningSchemeUpdateRequested and ProvisioningSchemeUpdateUntil fields on virtual machines, canceling any pending property updates. If a machine is already powering on with a property update, it cannot be cancelled with this cmdlet.

Examples

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

List of VM names. If not specified, all VMs in the machine catalog associated with the specified provisioning scheme will have pending property updates cancelled.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId' and 'ProvisioningSchemeName' to Cancel-ProvVMUpdate.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_ProvMachineCreationSnapin
- Clear-ProvVMUpdateTimeWindow

Clear-ProvSchemeWarning

March 11, 2024

Clears warnings from a Provisioning Scheme.

Syntax

```
Clear-ProvSchemeWarning
      -ProvisioningSchemeName <String>
      [-All]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Clear-ProvSchemeWarning
      -ProvisioningSchemeUid <Guid>
      [-All]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to clear warnings from a provisioning scheme.

Examples

EXAMPLE 1

Clears all warnings from the provisioning scheme named 'MyScheme'.

```
Get-ProvScheme -ProvisioningSchemeName MyScheme | Clear-
    ProvSchemeWarning -All
```

EXAMPLE 2

Clears all warnings from the provisioning scheme named 'MyScheme'.

```
Clear-ProvSchemeWarning -ProvisioningSchemeName MyScheme -All
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to clear warnings from.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme to clear warnings from.

Type:	Guid

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-All

Whether to clear all warnings from the provisioning scheme. This is the only option supported at this time.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe an object containing a string property called 'ProvisioningSchemeName'to Clear-ProvSchemeWarning.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin

Clear-ProvVMUpdateTimeWindow

March 11, 2024

Cancels pending property updates on provisioned virtual machines.

Syntax

```
Clear-ProvVMUpdateTimeWindow
     -ProvisioningSchemeName <String>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Clear-ProvVMUpdateTimeWindow
     -ProvisioningSchemeName <String>
     -VMName <String[]>
     [-LoggingId <Guid>]
```



```
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Clear-ProvVMUpdateTimeWindow
     -ProvisioningSchemeUid <Guid>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Clear-ProvVMUpdateTimeWindow
     -ProvisioningSchemeUid <Guid>
     -VMName <String[]>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Provides the ability to clear pending property updates on existing virtual machines.

This cmdlet clears the ProvisioningSchemeUpdateRequested and ProvisioningSchemeUpdateUntil fields on virtual machines, canceling any pending property updates. If a machine is already powering on with a property update, it cannot be cancelled with this cmdlet.

Examples

EXAMPLE 1

Clear the time window for all VMs in azure-catalog, canceling the pending property update for each machine.

```
Get-ProvVM -ProvisioningSchemeName azure-catalog | select VMName,
   ProvisioningSchemeVersion, ProvisioningSchemeUpdateRequested,
   ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
   ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
   ---------------------------------  -----------------------------
azu01                                1              3/11/2022  11:00:00
   PM         3/12/2022  12:00:00 AM
azu02                                1              3/11/2022  11:00:00
   PM         3/12/2022  12:00:00 AM
azu03                                1              3/11/2022  11:00:00
   PM         3/12/2022  12:00:00 AM
azu04                                1              3/11/2022  11:00:00
   PM         3/12/2022  12:00:00 AM

Clear-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog
```



```
10
11 Get-ProvVM -ProvisioningSchemeName azure-catalog | select VMName,
       ProvisioningSchemeVersion, ProvisioningSchemeUpdateRequested,
       ProvisioningSchemeUpdateUntil
12 VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
13 ------       -------------------------
       ---------------------------------  ----------------------------
14 azu01                                1
15 azu02                                1
16 azu03                                1
17 azu04                                1
```

EXAMPLE 2

Clear the time window for machines azu01,azu02 in azure-catalog, canceling the pending property update for the two machines. The other machines remain having a property update time window set.

```
 1 Get-ProvVM -ProvisioningSchemeName azure-catalog | select VMName,
       ProvisioningSchemeVersion, ProvisioningSchemeUpdateRequested,
       ProvisioningSchemeUpdateUntil
 2 VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
 3 ------       -------------------------
       ---------------------------------  ----------------------------
 4 azu01                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
 5 azu02                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
 6 azu03                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
 7 azu04                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
 8
 9 Cancel-ProvVMUpdate -ProvisioningSchemeName azure-catalog -VMName azu01
       ,azu02
10
11 Get-ProvVM -ProvisioningSchemeName azure-catalog | select VMName,
       ProvisioningSchemeVersion, ProvisioningSchemeUpdateRequested,
       ProvisioningSchemeUpdateUntil
12 VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
13 ------       -------------------------
       ---------------------------------  ----------------------------
14 azu01                                1
15 azu02                                1
16 azu03                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
17 azu04                                1               3/11/2022  11:00:00
       PM         3/12/2022  12:00:00 AM
```


Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

List of VM names. If not specified, all VMs in the machine catalog associated with the specified provisioning scheme will have pending property updates cancelled.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called ‘VMId’and ‘ProvisioningSchemeName’to Clear-ProvVMUpdateTimeWindow.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Set-ProvVMUpdateTimeWindow
- Set-ProvScheme

Confirm-ProvOperationEvent

March 11, 2024

Acknowledge a list of MCS operation events.

Syntax

```
Confirm-ProvOperationEvent
        [-EventId <Int32[]>]
        [-LinkedObjectType <String>]
        [-LinkedObjectUid <Guid>]
        [-All]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Allows you to set the EventState from New to Acknowledged.

Examples

EXAMPLE 1

Sets the operation event with a EventId of 42 to Acknowledged.

```
Set-ProvOperationEvent -EventId 42
```

EXAMPLE 2

Sets all operation events for provisioning scheme with ID 93f2c1b3-3015-4d63-92fe-678f1523b4de to Acknowledged.

```
Set-ProvOperationEvent -LinkedObjectType ProvisioningScheme -
    LinkedObjectUid 93f2c1b3-3015-4d63-92fe-678f1523b4de
```

Parameters

-EventId

The array of event id which all associated operation events will be set to Acknowledged.

Type:	Int32[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LinkedObjectType

The type of the linked object for which all associated operation events will be set to Acknowledged.

Type:	String
Accepted values:	ProvisioningScheme
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LinkedObjectUid

The unique identifier of the linked object for which all associated operation events will be set to Acknowledged.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-All

All operation events will be set to Acknowledged.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvOperationEvent
- Remove-ProvOperationEvent

Export-ProvScheme

March 11, 2024

Exports a provisioning scheme in the form of a JSON encoded string.

Syntax

```
Export-ProvScheme
      -ProvisioningSchemeName <String>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Export-ProvScheme
      -ProvisioningSchemeUid <Guid>
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The primary use of this cmdlet is to export a provisioning scheme and the VMs provisioned with this scheme. The exported JSON string can then be used to import this provisioning scheme and provisioned VMs into another site. The exported JSON string contains data about the provisioning scheme, image history, and provisioned VMs. However, it does not contain any AD accounts created during provisioning. Use cmdlet Export-AcctIdentityPool to export the identity pool containing those AD accounts.

Examples

EXAMPLE 1

Exports the provisioning scheme with the name MyScheme.

```
Export-ProvScheme -ProvisioningSchemeName MyScheme
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Export-ProvScheme.

Outputs

String

A JSON encoded string describing the provisioning scheme, image history, and VMs created with the provisioning scheme.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- Import-ProvScheme
- Export-AcctIdentityPool
- Import-AcctIdentityPool

Get-ProvDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the MachineCreation Service.

Syntax

```
Get-ProvDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected MachineCreation Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the MachineCreation Service instance running on controller "controller1.mydomain.net".

```
Get-ProvDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current MachineCreation Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the MachineCreationService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvDBConnection
- Get-ProvServiceStatus
- Test-ProvDBConnection

Get-ProvDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix MachineCreation Service.

Syntax

```
Get-ProvDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix MachineCreation Service database schema, add a new MachineCreation service to an existing site, remove a MachineCreation service from a site, or create a database server logon for a MachineCreation service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected MachineCreation service instance, otherwise the scripts relate to MachineCreation service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to MachineCreation service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to MachineCreation service roles

If ScriptType is Evict, the returned script contains:

- Removal of MachineCreation service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix MachineCreation Service and copies it to a file called "C:\MachineCreationSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a MachineCreation service schema.

```
Get-ProvDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\MachineCreationSchema.sql
```


EXAMPLE 2

Gets a script to create the appropriate database server logon for the MachineCreation service. This can be used when configuring a mirror server for use.

```
1  Get-ProvDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
       MachineCreationLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new MachineCreation service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the MachineCreation services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix MachineCreation Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a MachineCreation Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a MachineCreation Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the MachineCreation Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the

required permissions for local services to access the database schema for MachineCreation services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the MachineCreation Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvDBConnection
- Test-ProvDBConnection

Get-ProvDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix MachineCreation Service.

Syntax

```
Get-ProvDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix MachineCreation Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix MachineCreation Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix MachineCreation Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix MachineCreation Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix MachineCreation Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-ProvServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-ProvDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix MachineCreation Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-ProvDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix MachineCreation Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any MachineCreation services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-ProvServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvInstalledDBVersion
- Get-ProvServiceStatus
- Get-ProvDBSchema

Get-ProvImageDefinition

March 11, 2024

Gets the list of image definitions.

Syntax

```
Get-ProvImageDefinition
    [-PreparedImageDefinitionUid <Guid>]
    [[-PreparedImageDefinitionName] <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Lets you retrieve the list of defined image definitions.

Examples

EXAMPLE 1

Returns all of the available image definitions.

```
Get-ProvImageDefinition

CreateTime                      : 1/1/2022 12:00:00
CustomProperties                :
Description                     : Windows image
HostingUnitName                 : Hu
HostingUnitUid                  : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
OsType                          : Windows
PreparedImageDefinitionName     : Image1
PreparedImageDefinitionUid      : 7585f0de-192e-4847-a6d8-22713c3a2f42
UseWriteBackCache               : False

CreateTime                      : 1/1/2022 12:00:00
CustomProperties                :
Description                     : Linux image
HostingUnitName                 : Hu
HostingUnitUid                  : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
OsType                          : Linux
PreparedImageDefinitionName     : Image2
PreparedImageDefinitionUid      : b5a55401-ad36-4f53-b0e0-dbb6a5bdf6fd
UseWriteBackCache               : False
```


EXAMPLE 2

Returns all of the image definitions that have the name ‘Image0’or ‘Image1’.

```
Get-ProvImageDefinition -PreparedImageDefinitionName Image[0-1]

CreateTime                      : 1/1/2022 12:00:00
CustomProperties                :
Description                     : Windows image
HostingUnitName                 : Hu
HostingUnitUid                  : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
OsType                          : Windows
PreparedImageDefinitionName     : Image1
PreparedImageDefinitionUid      : 7585f0de-192e-4847-a6d8-22713c3a2f42
UseWriteBackCache               : False
```

Parameters

-PreparedImageDefinitionName

The name of the image definition.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PreparedImageDefinitionUid

The unique identifier of the image definition.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageDefinition

This object provides details of the image definition and contains the following information:

CreateTime <DateTime>

The date and time when the image definition was created.

CustomProperties <string>

Properties of the image definition which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

Description <string>

The description of the image definition.

HostingUnitName <string>

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the definition uses.

HostingUnitUid <Guid>

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the definition uses.

OsType <string>

The OS type for the image versions in this image definition.

PreparedImageDefinitionName <string>

The name of the image definition.

PreparedImageDefinitionUid <Guid>

The unique identifier for the image definition.

UseWriteBackCache <bool>

Whether or not to use write back cache for the image versions in this image definition.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvImageDefinition
- Remove-ProvImageDefinition
- Rename-ProvImageDefinition
- Set-ProvImageDefinition

Get-ProvImageRuntimeEnvironment

March 11, 2024

Get image runtime Environment.

Syntax

```
Get-ProvImageRuntimeEnvironment
   -ImageVersionUid <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-ProvImageRuntimeEnvironment
   -ProvisioningSchemeUid <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allow you to extract information from 'ImageRutimeInfo'field in provisioning scheme or image version.

Examples

EXAMPLE 1

Get the image runtime environment associated with the given image version.

```
$res = Get-ProvImageRuntimeEnvironment -ImageVersionUid a80c8338-e7af
    -48d6-8f08-21d2590b21b6
PS C:\Users\hanw> $res


Capabilities           : {
 [AzureAdSupported, True], [HybridAzureAdSupported, True], [
    IntuneSupported, False] }

DeviceEnrollmentStatus : Unknown
IdentityJoinStatus     : NotDomainJoined
OperatingSystem        : Citrix.MachineCreation.Sdk.OperatingSystemInfo
PagingFileSettings     : {
 Citrix.MachineCreation.Sdk.PagingFileSetting }

VDASessionSupport      : MultiSession



PS C:\Users\hanw> $res.OperatingSystem

FullName                          Type
--------                          ------------------
Windows Server 2019 Datacenter    Windows
```

EXAMPLE 2

Get the image runtime environment associated with the given provisioning scheme.

```
$res = Get-ProvImageRuntimeEnvironment -ProvisioningSchemeUid c0c55fa2
    -3d9f-44ee-8aba-3baec6547525
PS C:\Users\hanw> $res


Capabilities           : {
 [AzureAdSupported, False], [HybridAzureAdSupported, False], [
    IntuneSupported, True] }

DeviceEnrollmentStatus : Unknown
IdentityJoinStatus     : NotDomainJoined
OperatingSystem        : Citrix.MachineCreation.Sdk.OperatingSystemInfo
PagingFileSettings     :
VDASessionSupport      : SingleSession


```



```
15
16 PS C:\Users\hanw> $res.OperatingSystem
17
18 FullName       Type
19 --------       ------------------
20 Ubuntu 22.04   Linux
```

Parameters

-ImageVersionUid

The unique identifier of image version.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageRuntimeEnvironment

This object provides details of the image runtime environment and contains the following infrmation:

Capabilities <System.Collections.Generic.IDictionary<string, string»

The capabilities of the image such as 'AzureAdSupported'.

DeviceEnrollmentStatus <string>

The device enrollment status of the image.

IdentityJoinStatus <string>

The identity join status of the image.

OperatingSystem <Citrix.MachineCreation.Sdk.OperatingSystemInfo>

The operating system information of the image.

PagingFileSettings <Citrix.MachineCreation.Sdk.PagingFileSetting[]>

The page file settings of the image.

VDASessionSupport <string>

Specifies the session support of the VDA on the image. Valid values are: 'SingleSession', 'MultiSession', 'Unknown'.

Notes

The Get-ProvImageRuntimeEnvironment cmdlet returns an object containing the following six sub-objects:

- The image capabilities
- The device enrollment status
- The identity join status
- The operating system information of the image
- The page file settings of the image
- The type of VDA installed on the image

Related Links

- Get-ProvScheme
- Get-ProvImageVersion

Get-ProvImageScheme

March 11, 2024

Gets the list of image schemes.

Syntax

```
Get-ProvImageScheme
    [-PreparedImageSchemeUid <Guid>]
    [[-PreparedImageSchemeName] <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Lets you retrieve the list of defined image schemes.

Examples

EXAMPLE 1

Returns all of the available image schemes.

```
Get-ProvImageScheme

CpuCount                          : 1
CustomProperties                  :
HostingUnitName                   : Hu
HostingUnitUid                    : e5d4a2c9-6650-4cfc-93cc-1686db82b92e
MachineProfile                    :
MemoryMB                          : 1024
NetworkMaps                       : {
 0 }

PreparedImageSchemeName           : Scheme1
PreparedImageSchemeUid            : b5a55401-ad36-4f53-b0e0-dbb6a5bdf6fd
ServiceOffering                   :

CpuCount                          : 1
CustomProperties                  :
HostingUnitName                   : Hu
HostingUnitUid                    : e5d4a2c9-6650-4cfc-93cc-1686db82b92e
MachineProfile                    :
MemoryMB                          : 1024
NetworkMaps                       : {
 0 }

PreparedImageSchemeName           : Scheme2
PreparedImageSchemeUid            : 85bd21c6-851b-420a-a702-2fa2ee6f0052
ServiceOffering                   :
```

EXAMPLE 2

Returns all of the image schemes that have the name ‘Scheme0’or ‘Scheme1’.

```
Get-ProvImageScheme -PreparedImageSchemeName Scheme[0-1]

CpuCount                          : 1
CustomProperties                  :
HostingUnitName                   : Hu
HostingUnitUid                    : e5d4a2c9-6650-4cfc-93cc-1686db82b92e
MachineProfile                    :
MemoryMB                          : 1024
NetworkMaps                       : {
 0 }

PreparedImageSchemeName           : Scheme1
PreparedImageSchemeUid            : b5a55401-ad36-4f53-b0e0-dbb6a5bdf6fd
```



```
14 ServiceOffering                :
```

Parameters

-PreparedImageSchemeName

The name of the image scheme.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-PreparedImageSchemeUid

The unique identifier of the image scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageScheme

This object provides details of the image scheme and contains the following information:

CpuCount <int>

The number of processors that VMs will be created with when using this scheme.

CustomProperties <string>

Properties of the scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

HostingUnitName <string>

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme uses.

HostingUnitUid <Guid>

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the scheme uses.

MachineProfile <string>

The inventory path to the source VM used by the scheme as a template.

MemoryMB <int>

The maximum amount of memory that VMs will be created with when using this scheme.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

PreparedImageSchemeName <string>

The name of the image scheme.

PreparedImageSchemeUid <Guid>

The unique identifier for the image scheme.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData

Only a subset of the available data was returned.

CouldNotQueryDatabase

The query to get the database was not defined.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvImageScheme
- Remove-ProvImageScheme

Get-ProvImagesPendingDelete

March 11, 2024

Returns the list of pending image(s) to be deleted

Syntax

```
Get-ProvImagesPendingDelete
    [-HostingUnitUid <Guid>]
    [[-ImageId] <String>]
    [-StorageId <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns a list for each pending image which needs to be deleted. This check is done without regard for scoping of existing provisioning schemes, and hosting connection.

Examples

EXAMPLE 1

Gets the operation 'f68c0712-4643-496e-bf94-14b95b2e5c47'.

```
Get-ProvPendingImageDelete -HostingUnitUid "f68c0712-4643-496e-bf94-14
    b95b2e5c47"

HostingUnitUid : f68c0712-4643-496e-bf94-14b95b2e5c47
```



```
Id              : 1
ImageId         : 2cf68213-bd35-4adc-9c69-5de723cba469
InProgress      : True
PendingState    : DiskInUse
Reason          : Begin Delete threw an exception
StorageId       : 693fa4ab-1d67-478f-98a2-c4766d63d7eb
TimeAdded       : 8/25/2022 4:04:39 AM
```

Parameters

-ImageId

The identifiers of the image(s)/ to be tested.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-HostingUnitUid

The identifiers of the hosting units(s) to be tested.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StorageId

The identifier for the storage location on which the virtual image is located.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Host.Sdk.HostingUnit

Filter pending image for delete by hosting unit identifer.

Citrix.Host.Sdk.ImageId

Filter pending image for delete by image identifer.

Outputs

Citrix.MachineCreation.Sdk.PendingImageDelete

The object has the following properties: Id <Int> The unique image record id

HostingUnitUid <Guid> The unique identifier of the hostingunit.

ImageId <string> The unique identifier of the image to be deleted.

InProgress <bool> Indicates whether the operation is in progress.

StorageId <string> The storage location on the hypervisor where the image resides.

TimeAdded <DateTime> The date and time when the operation was last queued for action.

PendingState <string> The state of the pending images to be deleted.

Reason <string> The reason why the image is in the pending state for deletion.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned.

CouldNotQueryDatabase The query required to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM

Get-ProvImageVersion

March 11, 2024

Get a list of image versions.

Syntax

```
Get-ProvImageVersion
    [-PreparedImageVersionUid <Guid>]
    [[-PreparedImageVersionNumber] <String>]
    [-PreparedImageDefinitionUid <Guid>]
    [-PreparedImageDefinitionName <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to retrieve the list of defined image versions.

Examples

EXAMPLE 1

Returns all of the available image versions.

```
Get-ProvImageVersion

CreateTime                                : 1/1/2022 12:00:00
Description                               :
DiskSize                                  : 32
```



```
Error                                    :
HostingUnitUid                           : 0be23759-e276-4657-b863-
    e958a716cb6e
ImageRuntimeInfo                         :
ImageStatus                              : Prepared
ImageVersionMetadata                     :
MasterImageId                            : hu-dev-testing-rg/hu-dev-
    tsvda-snapshot
MasterImageVM                            : XDHyp:\HostingUnits\East US.
    region\image.folder\hu-dev-testing-rg.resourcegroup\hu-dev-tsvda-
    snapshot.snapshot
PageFileSettings                         : D:\pagefile.sys 0 0
PreparedImageDefinitionName              : myImage
PreparedImageDefinitionUid               : 42bd09f5-3fac-44a6-8b1c-7
    c09f1079cc0
PreparedImageReplicas                    :
PreparedImageSchemeUid                   : 53ed79c3-c771-4114-ba05-
    b999a62a836f
PreparedImageVersionNumber               : 1
PreparedImageVersionUid                  : b1c7602c-b080-4a63-846b-9
    d2429d54bd1
UpdatePageFileSettings                   : False
VMMetadata                               :
Warnings                                 : {
  }

WriteBackCacheDiskIndex                  : 0
WriteBackCacheDiskSize                   : 0
WriteBackCacheMemorySize                 : 0
```

EXAMPLE 2

Returns all of the available image versions for image definition “myImage”.

```
Get-ProvImageVersion -PreparedImageDefinitionName myImage


CreateTime                               : 1/1/2022 12:00:00
Description                              :
DiskSize                                 : 32
Error                                    :
HostingUnitUid                           : 0be23759-e276-4657-b863-
    e958a716cb6e
ImageRuntimeInfo                         :
ImageStatus                              : Prepared
ImageVersionMetadata                     :
MasterImageId                            : hu-dev-testing-rg/hu-dev-
    tsvda-snapshot
MasterImageVM                            : XDHyp:\HostingUnits\East US.
    region\image.folder\hu-dev-testing-rg.resourcegroup\hu-dev-tsvda-
    snapshot.snapshot
PageFileSettings                         : D:\pagefile.sys 0 0
```



```
PreparedImageDefinitionName                : myImage
PreparedImageDefinitionUid                 : 42bd09f5-3fac-44a6-8b1c-7
    c09f1079cc0
PreparedImageReplicas                      :
PreparedImageSchemeUid                     : 53ed79c3-c771-4114-ba05-
    b999a62a836f
PreparedImageVersionNumber                 : 1
PreparedImageVersionUid                    : b1c7602c-b080-4a63-846b-9
    d2429d54bd1
UpdatePageFileSettings                     : False
VMMetadata                                 :
Warnings                                   : {
  }

WriteBackCacheDiskIndex                    : 0
WriteBackCacheDiskSize                     : 0
WriteBackCacheMemorySize                   : 0
```

Parameters

-PreparedImageVersionNumber

The image version number to retrieve.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionUid

The image version identifier to retrieve.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionUid

The identifier of image definition where image versions are contained.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name of image definition where image versions are contained.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageVersion

This object provides details of the image version and contains the following information:

CreateTime <DateTime>

The date and time when the image version was created.

Description <string>

The description of the image version.

DiskSize <int>

The disk size (in GB) that is used to create VMs.

Error <string>

Error state that has occurred with this image version.

HostingUnitUid <Guid>

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the definition uses.

ImageRuntimeInfo <string>

The image runtime information like operating system info, VDA components, etc.

ImageStatus <string>

The status of the image version.

ImageVersionMetadata <string>

The metadata associated with this image version.

MasterImageId <string>

The hypervisor Id of master image used in this image version.

MasterImageVM <string>

The inventory path to the VM snapshot copy used in this image version.

PageFileSettings <string>

The page file settings to meet hypervisor-specific features demands.

PreparedImageDefinitionName <string>

The name of image definition used in this image version.

PreparedImageDefinitionUid <Guid>

The unique identifier of image definition used in this image version.

PreparedImageReplicas <Citrix.MachineCreation.Sdk.ImageReplica[]>

Image replication information for this image version.

PreparedImageSchemeUid <Guid>

The unique identifier of image scheme used in this image version.

PreparedImageVersionNumber <string>

The version number of the image version.

PreparedImageVersionUid <Guid>

The unique identifier for the image version.

UpdatePageFileSettings <bool>

Whether to update page file settings for write back cache

VMMetadata <string>

The metadata that will be used to created VMs in a plain text format.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this image version.

WriteBackCacheDiskIndex <int>

The disk index that the hypervisor reported as the actual disk index attachment point for the write back cache at image preparation time.

WriteBackCacheMemorySize <int>

The memory size in MB of write back cache in this image version.

WriteBackCacheDiskSize <int>

The disk size in MB of write back cache in this image version.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvImageVersion
- Remove-ProvImageVersion
- Set-ProvImageVersion

Get-ProvInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the MachineCreation Service.

Syntax

```
Get-ProvInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix MachineCreation Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-ProvDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix MachineCreation Service database schema version number.

```
Get-ProvInstalledDBVersion
```


EXAMPLE 2

Get the versions of the MachineCreation Service database schema for which upgrade scripts are supplied.

```
1  Get-ProvInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-ProvInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the MachineCreation

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvDBVersionChangeScript
- Get-ProvDBSchema

Get-ProvMetadataConfiguration

March 11, 2024

Get VM metadata configuration settings for a plugin.

Syntax

```
Get-ProvMetadataConfiguration
    [-PluginType <String>]
    [-ConfigurationName <String>]
    [-CitrixDefined <Boolean>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to get metadata settings for a specific hypervisor.

Examples

EXAMPLE 1

Get all configuration settings

```
Get-ProvMetadataConfiguration

PluginType ConfigurationName ConfigurationValue    CitrixDefined
---------- ----------------- ------------------    -------------
AzureRM    Extension         AADLoginForWindows             True
AzureRM    Extension         CustomScriptExtension         False
```

EXAMPLE 2

Get configuration settings for AzureRM plugin

```
Get-ProvMetadataConfiguration -PluginType "AzureRm"

PluginType ConfigurationName ConfigurationValue    CitrixDefined
---------- ----------------- ------------------    -------------
AzureRM    Extension         AADLoginForWindows             True
AzureRM    Extension         CustomScriptExtension         False
```


EXAMPLE 3

Get supported VM extensions for AzureRM plugin

```
Get-ProvMetadataConfiguration -PluginType "AzureRm" -ConfigurationName
    "Extension"

PluginType ConfigurationName ConfigurationValue    CitrixDefined
---------- ----------------- ------------------    -------------
AzureRM    Extension         AADLoginForWindows             True
AzureRM    Extension         CustomScriptExtension         False
```

Parameters

-PluginType

The name of the hypervisor plug-in factory. Currently, AzureRmFactory is the only supported plug-in factory.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ConfigurationName

The configuration name. Currently, Extension is the only supported configuration.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CitrixDefined

Indicates whether it is defined by Citrix

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvMetadataConfiguration

The object has the following properties:

PluginType <string>

The name of Plugin

ConfigurationName <string>

The name of configuration

ConfigurationValue <string>

The value of configuration

CitrixDefined <bool>

Indicates if the configuration is defined by Citrix.

Notes

In the case of failure, the following errors can result.

Error Codes

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error

DatabaseError An error occurred in the service while attempting a database operation.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

CommunicationError An error occurred while communicating with the service.

ExceptionThrown An unexpected error occurred. For more details, check the Windows event logs on your self-hosted delivery controller or contact Citrix support if using Citrix DaaS (Citrix-hosted delivery controller).

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvMetadataConfiguration
- Remove-ProvMetadataConfiguration

Get-ProvObjectReference

March 11, 2024

Returns the number of local objects holding references to objects from other services.

Syntax

```
Get-ProvObjectReference
   [-HostingUnitUid <Guid[]>]
   [-IdentityPoolUid <Guid[]>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns for each hosting unit or identity pool GUID the number of references by provisioning schemes or by long running task. This check is done without regard for scoping of existing provisioning schemes, references by inaccessible schemes are also checked.

Examples

EXAMPLE 1

This checks a single hosting unit for objects that have references to it. The result shows there is only one provisioning scheme relating to it.

```
Get-ProvObjectReference -HostingUnitUid 5B66A060-85E1-4DBD-9D1B-
   BF79881D3BB1

Count          : 1
ObjectId       : 5b66a060-85e1-4dbd-9d1b-bf79881d3bb1
Source         : ProvisioningScheme
Target         : HostingUnit

```



```
Count          : 0
ObjectId       : 5b66a060-85e1-4dbd-9d1b-bf79881d3bb1
Source         : Task
Target         : HostingUnit
```

EXAMPLE 2

This iterates all available identity pools for objects that have references to them, using specification by property name. The result shows that for each identity pool there is one provisioning scheme relating to it.

```
Get-AcctIdentityPool | Get-ProvObjectReference

Count          : 1
ObjectId       : 7a63cd15-8a6f-450e-9c09-72fb9f211898
Source         : ProvisioningScheme
Target         : IdentityPool

Count          : 0
ObjectId       : 7a63cd15-8a6f-450e-9c09-72fb9f211898
Source         : Task
Target         : IdentityPool
```

Parameters

-HostingUnitUid

The identifiers of the hosting units(s) to be tested.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IdentityPoolUid

The identifiers of the identity pool(s) to be tested.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Host.Sdk.HostingUnit

You can pipe an object containing a parameter called 'HostingUnitUid' to Get-ProvObjectReference.

Citrix.ADIdentity.Sdk.IdentityPool

You can pipe an object containing a parameter called 'IdentityPoolUid' to Get-ProvObjectReference.

Outputs

ObjectReferenceCount

An object containing the input object identifier, its type, the types of referencing objects, and the number of references.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin

Get-ProvOperationEvent

March 11, 2024

Gets a list of MCS operation events.

Syntax

```
Get-ProvOperationEvent
    [-EventId <Int32>]
    [-EventCategory <String>]
    [-EventSeverity <String>]
    [-EventSource <String>]
    [-OperationType <String>]
    [-OperationTargetType <String>]
    [-LinkedObjectType <String>]
    [-LinkedObjectUid <Guid>]
    [-Locale <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to retrieve the list of MCS operation events.

Examples

EXAMPLE 1

Returns the operation event with a Id of 42.

```
Get-ProvOperationEvent -EventId 42
```

EXAMPLE 2

Returns the operation events for provisioning scheme with ID 93f2c1b3-3015-4d63-92fe-678f1523b4de.

```
Get-ProvOperationEvent -LinkedObjectType ProvisioningScheme -
    LinkedObjectUid 93f2c1b3-3015-4d63-92fe-678f1523b4de
```

Parameters

-EventId

The id of the operation event to retrieve.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EventCategory

The category of the operation event to retrieve.

Type:	String
Accepted values:	Error, Warning
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EventSeverity

The severity of the operation event to retrieve.

Type:	String
Accepted values:	Critical, Important, Significant, Urgent
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EventSource

The source of the operation event to retrieve.

Type:	String
Accepted values:	AwsEc2Plugin, AzureRmPlugin, GcpPlugin, Mcs
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OperationType

The type of the operation event to retrieve.

Type:	String
Accepted values:	ConfigurationManagement, ImageManagement, None, PowerManagement, PreflightCheck, ProvisioningSchemeManagement, ProvisioningVmManagement, ResourceManagement
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OperationTargetType

The target type of the operation event to retrieve.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LinkedObjectType

The type of the linked object to get the operation events of.

Type:	String
Accepted values:	ProvisioningScheme
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LinkedObjectUid

The unique identifier of the linked object to get the operation events of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Locale

The locale to get the operation event message and recommendation of.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.OperationEvent

This object provides details of the operation event and contains the following information:

EventId <int> The unique identifier of this operation event.

LinkedObjectUid <Guid> The unique identifier for the linked resource this operation event is associated with.

LinkedObjectType <string> The type of the linked resource (e.g. ProvisioningScheme) this operation event is associated with.

EventCategory <string> The category of this operation event.

EventSeverity <string> The severity of this operation event.

EventSource <string> The source of this operation event.

EventState <string> The state of this operation event.

EventMessage <string> The description of this operation event.

EventDateTime <DateTime> The date and time this operation event was created (UTC).

EventAdditionalData <string> Optional, additional data related to this operation event.

OperationType <string> The description of this operation event.

OperationName <string> The description of this operation event.

OperationTargetType <string> The description of this operation event.

OperationTargetName <string> The description of this operation event.

Recommendation <string> Optional, the recommended action of this operation event.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvOperationEvent

Get-ProvOrphanedResource

March 11, 2024

Gets the list of orphaned resources created while schemes provisioning.

Syntax

```
Get-ProvOrphanedResource
   -HypervisorConnectionUid <Guid[]>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Let you retrieve the list of orphaned provisioning resources, such as disks, snapshots, galleries, virtual machines and network interfaces etc.

Examples

EXAMPLE 1

Returns the orphaned resource objects

```
$pluginId = 'AzureRmFactory'
$connections = Get-ChildItem xdhyp:\connections | where {
  $_.PluginId -eq $pluginId  }

$result = Get-ProvOrphanedResource -HypervisorConnectionUid
   $connections.HypervisorConnectionUid
$result
resourceId

   ResourceType              ProvisioningSchemeId
   Metadata
```



```
 8  -----------------

       ------------                --------
       --------
 9  {
10   /subscriptions/3fd5967f-7152-46c4-b061-2bd5d0cad70c/resourceGroups/
        citrix-xd-66575601-158f-48f1-a013-9c85e5f10c21-9peif/providers/
        Microsoft.Compute/snapshots/AZMC01-baseDisk-vy3ov }
11    microsoft.compute/disks    d5a53d4e-dc8b-416a-80ce-8eef5eb2256f
```

Parameters

-HypervisorConnectionUid

Connection Uids group to specific hypervisor.

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.HypResource

This object provides details of hypervisor orphaned resource.

Id <string> An Id of hypervisor orphaned resource.

ResourceType <string> Resource type of hypervisor orphaned resource.

ProvisioningSchemeId <string> ProvisioningSchemeId of hypervisor orphaned resource.

Metadata <Citrix.MachineCreation.Sdk.ProvisionedResourceMetadata[]> Hypervisor resource orphaned metadata key/value.

Notes

The Get-ProvOrphanedResource cmdlet returns an object containing three sub-objects:

- The Id of hypervisor orphaned resource.
- The ProvisioningSchemeId of hypervisor orphaned resource.
- The resource type of hypervisor orphaned resource.
- The hypervisor orphaned resource metadata.

Related Links

Get-ProvResource

March 11, 2024

Gets the list of resources created for provisioning schemes, image versions, and virtual machines.

Syntax

```
Get-ProvResource
   -ProvisioningSchemeName <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-ProvResource
   -ProvisioningSchemeUid <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Let you retrieve the list of provisioning resources, such as disks created for a provisioning scheme, image versions, and virtual machines.

Examples

EXAMPLE 1

Returns the ProvisionedScheme object, ProvisionedImageVersion object, and ProvisionedVirtualMachine object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
    a309-1d53f7882e92'
$result

ProvisionedScheme
    ProvisionedImageVersion
-----------------
    ----------------------
Citrix.MachineCreation.Sdk.ProvisionedSchemeResource {
 /example.vm/example.snapshot, /example...
```

EXAMPLE 2

Returns the Id, Name, and Resource of the ProvisionedScheme object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
    a309-1d53f7882e92'
$result.ProvisionedScheme

Id                                   Name                   Resource
--                                   ----                   --------
e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92 Example_Catalog
```

EXAMPLE 3

Returns ImageStatus, ID, and Name of the ProvisionedImageVersion sub-object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
    a309-1d53f7882e92'
$result.ProvisionedImageVersion

```



```
ImageStatus Id                                   Name
----------- --                                   ----
Deleted     92bd1cb2-4d16-43ad-8672-04b28889f072 /example.vm/example.
   snapshot
Current     def7badc-4769-4c5d-93e1-2c20018dca26 /example.vm/example.
   snapshot
Deleted     9dca4d28-acc6-49c8-bdc4-94cc12b958ea /example.vm/example.
   snapshot
```

EXAMPLE 4

Returns the Resource (Id, ResourceType, and Metadata) of the ProvisionedImageVersion object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
   a309-1d53f7882e92'
$result.ProvisionedImageVersion[1].Resource

Id                                   ResourceType Metadata
--                                   ------------ --------
0005f0fa-2f3f-4361-9103-5d9e6b9c656c Disk         {
 Role, StorageId }
```

EXAMPLE 5

Returns the Metadata (Role and StorageId) of the resource of the ProvisionedImageVersion sub-object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
   a309-1d53f7882e92'
$result.ProvisionedImageVersion[1].Resource[0].Metadata

Key       Value
---       -----
Role      Base Disk
StorageId 1a033f55-d552-5ba9-e364-2af7b859303e
```

EXAMPLE 6

Returns ID, Name, and Resource of the provisioned virtual machines for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
   a309-1d53f7882e92'
$result.ProvisionedVirtualMachine

```



```
Id                                   Name        Resource
--                                   ----        --------
7141f207-844f-f271-a076-10258186bb52 ExampleVM01 {
 00000000-0000-0000-0000-000000000000, 539e7ebb-296f-4021-9faa-8614
    ff159e77, 00000000-000...
1f8cceed-1c6b-efb3-079e-486477e2b2c2 ExampleVM02 {
 00000000-0000-0000-0000-000000000000, 2574798c-014c-4457-9040-
    d040246c2dc8, 00000000-000...
```

EXAMPLE 7

Returns the Resource (Id, ResourceType, and Metadata) of the ProvisionedVirtualMachine object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
    a309-1d53f7882e92'
$result.ProvisionedVirtualMachine[0].Resource

Id                                   ResourceType Metadata
--                                   ------------ --------
00000000-0000-0000-0000-000000000000 Disk         {
 Role, ParentDiskId, StorageId }

539e7ebb-296f-4021-9faa-8614ff159e77 Disk         {
 Role, StorageId }

00000000-0000-0000-0000-000000000000 Disk         {
 Role, StorageId }
```

EXAMPLE 8

Returns the Metadata (Role, ParentDiskID, and StorageId for OS Delta Disk) of the Resource of the ProvisionedVirtualMachine object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-
    a309-1d53f7882e92'
$result.ProvisionedVirtualMachine[0].Resource[0].Metadata

Key          Value
---          -----
Role         OS Delta Disk
ParentDiskId 5eef2ad5-2b05-442c-a5b5-1a7e9ecaf36d
StorageId    1a033f55-d552-5ba9-e364-2af7b859303e
```


EXAMPLE 9

Returns the Metadata (Role and StorageId for Identity Disk) of the Resource of the ProvisionedVirtualMachine object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92'
$result.ProvisionedVirtualMachine[0].Resource[1].Metadata

Key       Value
---       -----
Role      Identity Disk
StorageId 1a033f55-d552-5ba9-e364-2af7b859303e
```

EXAMPLE 10

Returns the Metadata (Role and StorageId for Personal Virtual Disk) of the Resource of the ProvisionedVirtualMachine object for the provisioning scheme ID "e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92."

```
$result = Get-ProvResource -ProvisioningSchemeUid 'e9aa1cf8-b2a8-4c3c-a309-1d53f7882e92'
$result.ProvisionedVirtualMachine[0].Resource[2].Metadata

Key       Value
---       -----
Role      PersonalVDisk
StorageId
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName' to Get-ProvResource.

Outputs

Citrix.MachineCreation.Sdk.ProvisionedResource

This object provides details of the provisioning scheme and contains the following information:

ProvisionedScheme <Citrix.MachineCreation.Sdk.ProvisionedSchemeResource>

An object containing provisioning scheme id, name, and resources attached to the provisioning scheme.

ProvisionedImageVersion <Citrix.MachineCreation.Sdk.ProvisionedImageVersionResource[] >

A list of objects containing provisioning scheme id, name, and resources attached to the provisioning image versions.

ProvisionedVirtualMachine <Citrix.MachineCreation.Sdk.ProvisionedVirtualMachineResource[] >

A list of objects containing provisioning scheme id, name, and resources attached to the provisioned virtual machines.

Notes

The Get-ProvResource cmdlet returns an object containing three sub-objects:

- The provisioned scheme
- Array of provisioned image versions
- Array of provisioned virtual machines

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- Get-ProvScheme
- Get-ProvSchemeMasterVMImageHistory
- Get-ProvVM

Get-ProvScheme

March 11, 2024

Gets a list of provisioning schemes.

Syntax

```
Get-ProvScheme
    [-ProvisioningSchemeUid <Guid>]
    [[-ProvisioningSchemeName] <String>]
    [-ScopeId <Guid>]
    [-ScopeName <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to retrieve the list of defined provisioning schemes.

Examples

EXAMPLE 1

Returns all of the available provisioning schemes.

```
Get-ProvScheme


CleanOnBoot                  : True
ControllerAddress            : {
  }

CpuCount                     : 1
DiskSize                     : 20
HostingUnitName              : HostUnit1
HostingUnitUid               : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName             : idPool1
IdentityPoolUid              : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                 : 0
MachineProfile               : XDHyp:/.../Profile.vm
MasterImageVM                : XDHyp:/.../Base.vm/base.snapshot
MasterImageVMDate            : 17/05/2020 09:27:50
MemoryMB                     : 1024
Metadata                     : {
 Department = Sales }

MetadataMap                  : {
 [Department = Sales] }
```



```
24
25 PreparedImageDefinitionName  :
26 PreparedImageVersionNumber   :
27 PreparedImageVersionUid      : 00000000-0000-0000-0000-000000000000
28 ProvisioningSchemeUid        : 7585f0de-192e-4847-a6d8-22713c3a2f42
29 ProvisioningSchemeVersion    : 1
30 TaskId                       : 00000000-0000-0000-0000-000000000000
31 VMMetadata                   : {
32  0, 1, 0, 0... }
33
34 PersonalVDiskDriveLetter     :
35 PersonalVDiskDriveSize       : 0
36 UsePersonalVDiskStorage      : False
37 NetworkMaps                  : {
38  0 }
39
40 Scopes                       :
41 DedicatedTenancy             : False
42 GpuTypeId                    :
43 ResetAdministratorPasswords  : False
44 SecurityGroups               : {
45   }
46
47 ServiceOffering              :
48 TenancyType                  : Shared
49 AzureAdJoinType              :
50 CurrentMasterImageUid        : c0571690-4f57-4476-901b-fe64d6aecb79
51 CustomProperties             :
52 IdentityType:                : ActiveDirectory
53 UseFullDiskCloneProvisioning : False
54 UseWriteBackCache            : True
55 WriteBackCacheDiskSize       : 24
56 WriteBackCacheMemorySize     : 256
57 Warnings                     : {
58   }
59
60 WriteBackCacheDiskIndex     : 0
61 WindowsActivationType        : MultipleActivationKey
62
63 CleanOnBoot                  : True
64 ControllerAddress            : {
65   }
66
67 CpuCount                     : 1
68 DiskSize                     : 20
69 HostingUnitName              : HostUnit1
70 HostingUnitUid               : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
71 IdentityPoolName             : idPool1
72 IdentityPoolUid              : 03743136-e43b-4a87-af74-ab71686b3c16
73 MachineCount                 : 100
74 MachineProfile               : XDHyp:/.../Profile.vm
75 MasterImageVM                : XDHyp:/.../Base.vm/base.snapshot
76 MasterImageVMDate            : 17/05/2020 09:27:50
```



```
MemoryMB                       : 1024
Metadata                       : {
  }

MetadataMap                    : {
  }

ProvisioningSchemeUid          : 43d82099-1fd7-4617-93f0-25b160813905
ProvisioningSchemeName         : Scheme2
ProvisioningSchemeVersion      : 1
TaskId                         : 00000000-0000-0000-0000-000000000000
VMMetadata                     : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter       :
PersonalVDiskDriveSize         : 0
UsePersonalVDiskStorage        : False
NetworkMaps                    : {
 0 }

Scopes                         :
DedicatedTenancy               : False
GpuTypeId                      :
ResetAdministratorPasswords    : False
SecurityGroups                 : {
  }

ServiceOffering                :
TenancyType                    : Shared
AzureAdJoinType                :
CurrentMasterImageUid          : 022cd6e4-34cb-3f7c-e02a-44ac404483b4
CustomProperties               :
IdentityType:                  : ActiveDirectory
UseFullDiskCloneProvisioning   : False
UseWriteBackCache              : True
WriteBackCacheDiskSize         : 24
WriteBackCacheMemorySize       : 256
Warnings                       : {
  }

WriteBackCacheDiskIndex        : 0
WindowsActivationType          : MultipleActivationKey
```

EXAMPLE 2

Returns all of the provisioning schemes that have the name ‘Scheme0’or ‘Scheme1’.

```
Get-ProvScheme -ProvisioningSchemeName Scheme[0-1]

CleanOnBoot                    : True
ControllerAddress              : {
  }
```



```
6
7  CpuCount                      : 1
8  DiskSize                      : 20
9  HostingUnitName               : HostUnit1
10 HostingUnitUid                : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
11 IdentityPoolName              : idPool1
12 IdentityPoolUid               : 03743136-e43b-4a87-af74-ab71686b3c16
13 MachineCount                  : 0
14 MachineProfile                : XDHyp:/.../Profile.vm
15 MasterImageVM                 : XDHyp:/.../Base.vm/base.snapshot
16 MasterImageVMDate             : 17/05/2020 09:27:50
17 MemoryMB                      : 1024
18 Metadata                      : {
19  Department = Sales }
20
21 MetadataMap                   : {
22  [Department = Sales] }
23
24 ProvisioningSchemeUid         : 7585f0de-192e-4847-a6d8-22713c3a2f42
25 ProvisioningSchemeName        : Scheme1
26 ProvisioningSchemeVersion     : 1
27 TaskId                        : 00000000-0000-0000-0000-000000000000
28 VMMetadata                    : {
29  0, 1, 0, 0... }
30
31 PersonalVDiskDriveLetter      :
32 PersonalVDiskDriveSize        : 0
33 UsePersonalVDiskStorage       : False
34 NetworkMaps                   : {
35  0 }
36
37 Scopes                        :
38 DedicatedTenancy              : False
39 GpuTypeId                     :
40 ResetAdministratorPasswords   : False
41 SecurityGroups                : {
42   }
43
44 ServiceOffering               :
45 TenancyType                   : Shared
46 AzureAdJoinType               :
47 CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
48 CustomProperties              :
49 IdentityType:                 : ActiveDirectory
50 UseFullDiskCloneProvisioning : False
51 UseWriteBackCache             : True
52 WriteBackCacheDiskSize        : 24
53 WriteBackCacheMemorySize      : 256
54 Warnings                      : {
55   }
56
57 WriteBackCacheDiskIndex       : 0
58 WindowsActivationType         : MultipleActivationKey
```


Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ScopeId

Gets only results with a scope matching the specified scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ScopeName

Gets only results with a scope matching the specified scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

CleanOnBoot <bool>

Indicates whether the VMs created will be reset to a clean state on each start.

ControllerAddress <string[]>

The DNS names of the controllers associated with this provisioning scheme.

CpuCount <int>

The number of processors that will be used to create VMs.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

HostingUnitName <string>

The name of the hosting unit being used by this provisioning scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this provisioning scheme.

IdentityPoolName <string>

The name of the identity pool being used by this provisioning scheme.

IdentityPoolUid <Guid>

The unique identifier of the identity pool being used by this provisioning scheme.

MachineCount <int>

The count of machines created with this provisioning scheme.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

MasterImageVM <string>

The inventory path to the VM snapshot copy used by this provisioning scheme.

MasterImageVMDate <DateTime>

The date and time when the VM snapshot copy used by this provisioning scheme was made.

MemoryMB <int>

The maximum amount of memory that will be used to created VMs in MB.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The metadata associated with this provisioning scheme.

MetadataMap <IDictionary[string, string];>

The metadata associated with this provisioning scheme arranged in key value pairs.

PreparedImageDefinitionName <string>

The name for the image definition used for the provisioning scheme.

PreparedImageVersionNumber <string>

The version number for the image version used for the provisioning scheme.

PreparedImageVersionUid <Guid>

The identifier for the image version used for the provisioning scheme.

ProvisioningSchemeName <string>

The name of this provisioning scheme.

ProvisioningSchemeUid <Guid>

The unique identifier for this provisioning scheme.

ProvisioningSchemeVersion <int>

The version of the provisioning scheme.

TaskId <Guid>

The identifier of any current task that is running for the provisioning scheme.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

PersonalVDiskDriveLetter <char>

The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int>

The size of the personal vDisk in GB.

UsePersonalVDiskStorage <bool>

Indicates whether this provisioning scheme uses personal vDisk storage.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

Scope <Citrix.MachineCreation.Sdk.ScopeReference[]>

The administration scopes associated with this provisioning scheme.

DedicatedTenancy <bool>

Indicates whether dedicated tenancy is used when creating VMs in Cloud Hypervisors.

GpuTypeId <string>

The id of the GPU (Graphics Processor Unit) Type used by this scheme. It is null if there is no GPU.

ResetAdministratorPasswords <bool>

Indicates whether the passwords for administrator accounts are reset on created machines.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

TenancyType <string>

Tenancy type to be used when creating VMs in Cloud Hypervisors. (See New-ProvScheme.)

AzureAdJoinType <string>

Deprecated.

CurrentMasterImageUid <Guid>

The unique identifier of the current master image used by the provisioning scheme. (See Get-ProvSchemeMasterVMImageHistory.)

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

IdentityType <string>

Identity type used to join created machines to a directory service. (See New-ProvScheme.)

UseFullDiskCloneProvisioning <bool>

Indicates whether VMs will be created using the dedicated full disk clone feature.

UseWriteBackCache <bool>

Indicates whether this provisioning scheme will use the write-back cache feature.

WriteBackCacheDiskSize <int>

The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int>

The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

WindowsActivationType <string>

Windows Activation Type set on the Master Image which has a mapping with the Provisioning Scheme.This feature is supported from 2303 and successive VDA versions. Any previous VDA version or if the vda is not of Windows Operating System Type, the field would be "UnsupportedVDA"

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData

Only a subset of the available data was returned

CouldNotQueryDatabase

The query to get the database was not defined.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvScheme
- Remove-ProvScheme
- Add-ProvSchemeMetadata
- Remove-ProvSchemeMetadata
- Add-ProvSchemeControllerAddress
- Remove-ProvSchemeControllerAddress

Get-ProvSchemeMasterVMImageHistory

March 11, 2024

Gets the list of master VM snapshots that have been used to provide hard disks to provisioning schemes.

Syntax

```
Get-ProvSchemeMasterVMImageHistory
   [-ProvisioningSchemeName <String>]
   [-ProvisioningSchemeUid <Guid>]
   [-MasterImageVM <String>]
   [-VMImageHistoryUid <Guid>]
   [-ImageStatus <String>]
   [-ShowAll]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Provides the ability to discover the master VM snapshots used to provide the hard disk image for provisioning schemes. This information includes the date and time when the image was created.

By default, this cmdlet returns only those snapshots that have been used previously. This information can be used to roll-back a provisioning scheme to a previous image.

The ShowAll parameter may be supplied to also show the snapshot for the image currently in use, and the snapshots for any images prepared for later use.

Examples

EXAMPLE 1

Gets all the hard disk images that have been used across all provisioning schemes.

```
Get-ProvSchemeMasterVMImageHistory


VMImageHistoryUid      : 3cba3a75-89cd-4868-989b-27feb378fec5
ProvisioningSchemeUid  : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : MyScheme
MasterImageVM          : /Base.vm/base.snapshot
Date                   : 17/05/2020 09:27:50
FunctionalLevel        :
ImageStatus            : Current
MasterImageId          : base-vm\base
MasterImageNote        : Office365 installed
```

EXAMPLE 2

Roll back the provisioning scheme to use the hard disk from the update1.snapshot for the provisioning scheme called “MyScheme”.

```
Get-ProvSchemeMasterVMImageHistory -ProvisioningSchemeName MyScheme -
    masterImageVM "/BaseXp.vm/update1.snapshot" | Publish-
    ProvMasterVMImage
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-MasterImageVM

The inventory path to the VM snapshot item used as the master VM image.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-VMImageHistoryUid

The unique identifier of the image history item.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ImageStatus

The status of the provisioning scheme image.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ShowAll

Show all images. This includes images currently in use, and images prepared for later use.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.VMImage

This object represents a master VM image history item. It contains the following parameters:

VMImageHistoryUid <Guid>

The unique identifier of the history item.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme used for the VM.

ProvisioningSchemeName <string>

The name of the provisioning scheme used for the VM.

MasterImageVM <string>

The inventory path to the Snapshot item that was used as the master VM image.

Date <DateTime>

The date and time when the VM or snapshot was used in the provisioning scheme.

FunctionalLevel <string>

The FunctionalLevel of the VDA installed on the VM.

ImageStatus <string>

The status of the provisioning scheme image.

MasterImageId <string>

The unique identifier of the snapshot, as assigned by the hosting unit.

MasterImageNote <string>

The note of the provisioning scheme image.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData

Only a subset of the available data was returned.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CouldNotQueryDatabase

The query required to get the database was not defined.

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Publish-ProvMasterVMImage
- Remove-ProvSchemeMasterVMImageHistory

Get-ProvSchemeResourceInStorage

March 11, 2024

Gets the summary of the base disk and machine count in each OS storage for the provisioning scheme.

Syntax

```
Get-ProvSchemeResourceInStorage
   [-ProvisioningSchemeName] <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-ProvSchemeResourceInStorage
   -ProvisioningSchemeUid <Guid>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Lets you retrieve the resource summary in each OS storage of the provisioning scheme, such as base disks and the machine count.

Examples

EXAMPLE 1

Returns the provisioning scheme name, hosting unit name, hosting unit id, and provisioning resource in storage object of the associated provisioning scheme id.

```
Get-ProvSchemeResourceInStorage -ProvisioningSchemeUid '9c08beec-154e-4
   a00-ad56-f97adb69aad4'

  ProvisioningSchemeName       : Vmware-MC
```



```
 ProvisioningSchemeUid         : 9c08beec-154e-4a00-ad56-f97adb69aad4
 HostingUnitName               : Vmware-hostingUnit
 HostingUnitUid                : 56b22b5e-b54d-49bc-a6f7-5dd336f6f70b
 ProvResourceInStorage         : {
datastore-11, datastore-12 }
```

EXAMPLE 2

Returns the provisioning scheme name, hosting unit name, hosting unit id, and provisioning resource in storage object of all the provisioning schemes with hosting unit as “Vmware-hostingUnit”.

```
Get-ProvScheme |Where-Object {
 $_.HostingUnitName -eq "Vmware-hostingUnit" }
  | Get-ProvSchemeResourceInStorage

 ProvisioningSchemeName        : Vmware-MC
 ProvisioningSchemeUid         : 9c08beec-154e-4a00-ad56-f97adb69aad4
 HostingUnitName               : Vmware-hostingUnit
 HostingUnitUid                : 56b22b5e-b54d-49bc-a6f7-5dd336f6f70b
 ProvResourceInStorage         : {
datastore-11, datastore-12 }


 ProvisioningSchemeName        : Vmware-MC2
 ProvisioningSchemeUid         : e9aa1cf8-b2a8-4c3c-a309-1d53f7882e76
 HostingUnitName               : Vmware-hostingUnit
 HostingUnitUid                : 56b22b5e-b54d-49bc-a6f7-5dd336f6f70b
 ProvResourceInStorage         : {
datastore-11, datastore-12 }
```

EXAMPLE 3

Returns the ID, name, and resource of the provisioned scheme object of the provisioning scheme id “9c08beec-154e-4a00-ad56-f97adb69aad4”.

```
$result = Get-ProvSchemeResourceInStorage -ProvisioningSchemeUid '9
    c08beec-154e-4a00-ad56-f97adb69aad4'
$result.ProvResourceInStorage | Format-List -Property *

OSStorageId         : datastore-11
OSStoragePath       : /Datacenter1.datacenter/Cluster1.cluster/
    datastore1.storage
OSStorageSuperseded : False
BaseDiskId          : {
 Version=2,LsiLogicScsi,1000,group-v4,[Datastore1]
Vmware-MC-baseDisk-datastore-11/Vmware-MC-baseDisk-datastore-11.vmdk,
    resgroup-93,Vmfs }

MachineCount        : 2
```



```
12
13 OSStorageId          : datastore-12
14 OSStoragePath        : /Datacenter1.datacenter/Cluster1.cluster/
      datastore2.storage
15 OSStorageSuperseded : False
16 BaseDiskId           : {
17   }
18
19 MachineCount         : 0
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName' to Get-ProvSchemeResourceInSt

Outputs

Citrix.MachineCreation.Sdk.ProvSchemeResourceInStorage

The object provides the resource details of the provisioning scheme and contains the following information:

ProvisioningSchemeName <string>

The name of the provisioning scheme.

ProvisioningSchemeUid <Guid>

The unique identifier for the provisioning scheme.

HostingUnitName <string>

The name of the hosting unit being used by the provisioning scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by the provisioning scheme.

ProvisionedResourceInStorage <Citrix.MachineCreation.Sdk.ProvisionedResourceInStorage[]>

A list of objects containing the OS storage id, path, base disk Id, and the number of machines located in the storage.

Notes

The Get-ProvSchemeResourceInStorage cmdlet returns an object containing the following five sub-objects:

- The provisioned scheme name
- The provisioned scheme id
- The hosting unit name
- The hosting unit id
- Array of provisioned resource in OS storage

Related Links

- Get-ProvScheme
- Get-ProvResource

Get-ProvSchemeVersion

March 11, 2024

Gets saved provisioning scheme configuration versions

Syntax

```
Get-ProvSchemeVersion
    [-ProvisioningSchemeUid] <Guid>
    [-Version <Int32>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Get-ProvSchemeVersion
    [-ProvisioningSchemeName] <String>
    [-Version <Int32>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
```



```
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to list saved provisioning scheme configuration versions.

Examples

EXAMPLE 1

Configurations are listed for provisioning scheme AzureCatalog

```
Get-ProvSchemeVersion -ProvisioningSchemeName AzureCatalog
CpuCount                : 2
CustomProperties        :
MachineProfile          : machineprofile.folder\matspo.resourcegroup\
   matspo-vm.templatespec\hibernate-2nic-fix.templatespecversion
MemoryInMB              : 14336
NetworkMaps             : {
 0, 1 }

ProvisioningSchemeName : AzureCatalog
ProvisioningSchemeUid  : 5b37b311-fa3f-49dd-b93b-661b9e6fa571
SecurityGroups          : {
  }

ServiceOffering         : serviceoffering.folder\Standard_D2s_v5.
   serviceoffering
VMMetadata              : {
 , , ,  ... }

Version                 : 2

CpuCount                : 2
CustomProperties        :
MachineProfile          : machineprofile.folder\matspo.resourcegroup\
   matspo-vm.templatespec\hibernate-2nic-fix.templatespecversion
MemoryInMB              : 14336
NetworkMaps             : {
 0, 1 }

ProvisioningSchemeName : AzureCatalog
ProvisioningSchemeUid  : 5b37b311-fa3f-49dd-b93b-661b9e6fa571
SecurityGroups          : {
  }

```



```
ServiceOffering        : serviceoffering.folder\Standard_D2s_v5.
    serviceoffering
VMMetadata             : {
 , , ,  ... }

Version                : 1
```

EXAMPLE 2

Version 2 is listed for provisioning scheme AzureCatalog

```
Get-ProvSchemeVersion -ProvisioningSchemeName AzureCatalog -Version 2
CpuCount               : 2
CustomProperties       :
MachineProfile         : machineprofile.folder\matspo.resourcegroup\
    matspo-vm.templatespec\hibernate-2nic-fix.templatespecversion
MemoryInMB             : 14336
NetworkMaps            : {
 0, 1 }

ProvisioningSchemeName : AzureCatalog
ProvisioningSchemeUid  : 5b37b311-fa3f-49dd-b93b-661b9e6fa571
SecurityGroups         : {
  }

ServiceOffering        : serviceoffering.folder\Standard_D2s_v5.
    serviceoffering
VMMetadata             : {
 , , ,  ... }

Version                : 2
```

Parameters

-ProvisioningSchemeUid

Filter by the unique identifier of the provisioning scheme.

Type:	Guid
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeName

Filter by the name of the provisioning scheme.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-Version

Filter by the version of the provisioning scheme

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningSchemeVersion

The object has the following properties:

Version <int>

The current version of the configuration

CpuCount <int>

The number of processors allocated to VMs in the provisioning scheme.

MemoryInMB <int>

The maximum amount of memory allocated to VMs in the provisioning scheme.

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in cloud hypervisors.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the VM.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the VM.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvScheme
- Remove-ProvSchemeVersion

Get-ProvSchemeWarning

March 11, 2024

Gets a list of provisioning schemes warnings.

Syntax

```
Get-ProvSchemeWarning
   [-ProvisioningSchemeUid <Guid>]
   [-ProvisioningSchemeName <String>]
   [-WarningId <Int32>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to retrieve the list of provisioning scheme warnings.

Examples

EXAMPLE 1

Returns the warning with a warning Id of 42.


```
Get-ProvSchemeWarning -WarningId 42

ProvisioningSchemeName       : CatalogWindows11AWS
ProvisioningSchemeUid        : 7585f0de-192e-4847-a6d8-22713c3a2f42
WarningId                    : 42
WarningMessage               : Deprecated custom attribute
RecommendedAction            : Remove obsolete custom attribute
AdditionalData               : AwsCaptureInstanceProperties is now
   deprecated
WarningStateType             : New
OperationName                : New-ProvScheme
WarningReportingDateTime     : 17/05/2022 09:27:50
```

Parameters

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme to get the warnings of.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeName

The name of the provisioning scheme to get the warnings of.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-WarningId

The id of the warning to retrieve.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvWarning

This object provides details of the provisioning scheme warning and contains the following information:

WarningId <int> The Unique identifier of this warning.

ProvisioningSchemeUid <Guid> The unique identifier for this provisioning scheme this warning is associated with.

ProvisioningSchemeName <Guid> The name for this provisioning scheme this warning is associated with.

WarningMessage <string[]> The warning description.

RecommendedAction <string> The optional recommended action to take to help resolve the warning.

AdditionalData <string> Optional, additional data related to the warning.

WarningStateType <string> The warning state of this provisioning scheme warning, either New or Acknowledged.

OperationName <string> The operation which triggered the warning.

WarningReportingDateTime <DateTime> The date and time the warning was created (UTC).

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvSchemeWarning
- Set-ProvSchemeWarning

Get-ProvScopedObject

March 11, 2024

Gets the details of the scoped objects for the Machine Creation Service.

Syntax

```
Get-ProvScopedObject
   [-ScopeId <Guid>]
   [-ScopeName <String>]
   [-ObjectType <ScopedObjectType>]
   [-ObjectId <String>]
   [-ObjectName <String>]
   [-Description <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a list of directly scoped objects including the names and identifiers of both the scope and object as well as the object description for display purposes.

There will be at least one result for every directly scoped object. When an object is associated with multiple scopes the output contains one result per scope duplicating the object details.

No records are returned for the All scope, though if an object is not in any scope a result with a null ScopeId and ScopeName will be returned.

Examples

EXAMPLE 1

Gets all of the scoped objects with type Scheme. The example output shows a scheme object (MyExampleScheme) in two scopes Sales and Finance, and another scheme (AnotherScheme) that is not in any scope. The ScopeId and ScopeName values returned are null in the final record.

```
Get-ProvScopedObject -ObjectType Scheme

ScopeId     : eff6f464-f1ee-4442-add3-99982e0cec01
ScopeName   : Sales
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     : 304e0fa7-d390-47f0-a94f-7e956a324c41
ScopeName   : Finance
ObjectType  : Scheme
ObjectId    : cd4174ee-9e4b-4e57-b126-9dbf757fe493
ObjectName  : MyExampleScheme
Description : Test scheme

ScopeId     :
ScopeName   :
ObjectType  : Scheme
ObjectId    : 5062e46b-71bc-4ac9-901a-30fe6797e2f6
ObjectName  : AnotherScheme
Description : Another scheme in no scopes
```

Parameters

-ScopeId

Gets scoped object entries for the given scope identifier.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ScopeName

Gets scoped object entries with the given scope name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectType

Gets scoped object entries for objects of the given type.

Type:	ScopedObjectType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ObjectId

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ObjectName

Gets scoped object entries for objects with the specified object identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Description

Gets scoped object entries for objects with the specified description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ScopedObject

You can pipe any object containing a parameter called 'ScopeId', 'ScopeName', 'ObjectType', 'ObjectId' or 'ObjectName', or any combination of them as filter parameter(s) to Get-ProvScopedObject.

Outputs

Citrix.MachineCreation.Sdk.ScopedObject

The Get-ProvScopedObject command returns an object containing the following properties:

ScopeId <Guid?>

Unique identifier of the scope.

ScopeName <string>

Display name of the scope.

ObjectType <ScopedObjectType>

Type of the object this entry relates to.

ObjectId <string>

Unique identifier of the object.

ObjectName <string>

Display name of the object.

Description <string>

Description of the object (possibly $null if the object type does not have a description).

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin

Get-ProvService

March 11, 2024

Gets the service record entries for the MachineCreation Service.

Syntax

```
Get-ProvService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Returns instances of the MachineCreation Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the MachineCreation Service running in the current service group.

```
Get-ProvService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries. The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.Service

The Get-ProvServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin

Get-ProvServiceAddedCapability

March 11, 2024

Gets any added capabilities for the MachineCreation Service on the controller.

Syntax

```
Get-ProvServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the MachineCreation Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the MachineCreation Service.

```
Get-ProvServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin

Get-ProvServiceConfigurationData

March 11, 2024

Gets configuration data for the service.

Syntax

```
Get-ProvServiceConfigurationData
    [-Name <String[]>]
    [-Value <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
```



```
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to retrieve the configuration parameters for the Machine Creation Service.

Examples

EXAMPLE 1

Get the Configuration data for the service.

```
Get-ProvConfigurationData

Name                : DeltaDiskDelete.timeDelay
Value               : 10
```

Parameters

-Name

The configuration data item name.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Value

The configuration data item value.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ServiceConfigurationData

This object provides details of the configuration data and contains the following information:

Name <string>

The name for the configuration item.

Value <string>

The value for the configuration item.

Notes

In the case of failure the following errors can be produced.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvServiceConfigurationData
- Remove-ProvServiceConfigurationData

Get-ProvServiceInstance

March 11, 2024

Gets the service instance entries for the MachineCreation Service.

Syntax

```
Get-ProvServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the MachineCreation Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the MachineCreation Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
1 Get-ProvServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ServiceInstance

The Get-ProvServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Prov.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvServiceStatus
- Reset-ProvServiceGroupMembership

Get-ProvServiceStatus

March 11, 2024

Gets the current state of the MachineCreation Service on the controller.

Syntax

```
Get-ProvServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables the status of the MachineCreation Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the MachineCreation Service.

```
Get-ProvServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-ProvServiceStatus command returns an object containing the status of the MachineCreation Service together with extra diagnostics information.

DBUnconfigured

The MachineCreation Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the MachineCreation Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the MachineCreation Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The MachineCreation is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The MachineCreation is connected to a database that is valid, but it does not have the full functionality required so the MachineCreation cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the MachineCreation Service currently in use is incompatible with the version of the MachineCreation Service schema on the database. Upgrade the MachineCreation Service to a more recent version.

DBOlderVersionThanService

The version of the MachineCreation Service schema on the database is incompatible with the version of the MachineCreation Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The MachineCreation Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the MachineCreation Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the MachineCreation and the database has been lost for an extended period of time, or has failed due to a configuration problem. The MachineCreation service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvDBConnection
- Test-ProvDBConnection
- Get-ProvDBConnection
- Get-ProvDBSchema

Get-ProvTask

March 11, 2024

Gets the task history for the Machine Creation Service.

Syntax

```
Get-ProvTask
   [[-TaskId] <Guid>]
   [-Type <JobType>]
   [-Active <Boolean>]
   [-Metadata <String>]
```



```
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns a list of tasks that have run or are currently running on the Machine Creation Service.

Examples

EXAMPLE 1

Get all completed tasks for the Machine Creation Service. All tasks will publish at least the fields listed above, plus more related to the particular task being performed.

```
Get-ProvTask -Active $false

TaskId                   : cfe5b3a2-c471-443e-b05e-8658e672d10f
Active                   : False
Host                     : NYSERVER
DateStarted              : 10/10/2020 15:27:34
Type                     : DisusedImageCleanup
Metadata                 : {
  }

CurrentOperation         :
TaskProgress             : 100
LastUpdateTime           : 10/10/2020 15:28:12
ActiveElapsedTime        : 56
DateFinished             : 10/10/2020 15:28:12
TerminatingError         :
Status                   : Finished
TaskExpectedCompletion   : 10/10/2012 15:28:12
...
```

Parameters

-TaskId

Specifies the task identifier to be returned.

Type:	Guid
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Type

Specifies the type of task to be returned.

Type:	JobType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Active

Specifies whether currently running tasks only or completed tasks only are returned.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Specifies the metadata entry that returned tasks should contain.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains at least the fields listed below. The fields output order and additional fields are related to the particular task type being performed.

TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Type <Citrix.XDInterServiceTypes.JobType>

The type of the task.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored against the task. For new tasks this will be empty until metadata is added.

CurrentOperation <string>

Operation specific phase of the overall "Running"task state.

TaskProgress <double>

The progress of the task 0-100%.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Status <string>

Where in its lifecycle the task is.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

Notes

If the command fails, the following errors can result.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvTask
- Stop-ProvTask
- Switch-ProvTask
- Add-ProvTaskMetadata
- Remove-ProvTaskMetadata

Get-ProvVM

March 11, 2024

Gets the VMs created using Machine Creation Services.

Syntax

```
Get-ProvVM
    [-ProvisioningSchemeUid <Guid>]
    [[-ProvisioningSchemeName] <String>]
    [-VMName <String>]
    [-Locked <Boolean>]
    [-Tag <String>]
    [-OutOfDate]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to obtain a list of the VMs that were created using Citrix Machine Creation Services.

Examples

EXAMPLE 1

Gets all the Virtual Machines that were provisioned using the Provisioning Scheme called 'MyScheme'.

```
Get-ProvVM -provisioningSchemeName MyScheme

ADAccountName                    : MYDomain\computer2$
ADAccountSid                     : S
   -1-5-21-3751941309-1176885247-1409628468-3179
CpuCount                         : 1
CreationDate                     : 17/05/2021 09:35:22
Domain                           : steve.dum.local
ImageOutOfDate                   : False
Lock                             : True
MemoryMB                         : 512
ProvisioningSchemeName           : XenPS
ProvisioningSchemeUid            : 5135a865-ba49-4e5f-87f2-2
   d65ee7a4e51
ProvisioningSchemeUpdateRequested : 17/05/2021 09:39:03
ProvisioningSchemeVersion        : 1
Tag                              : Brokered
```



```
VMId                                : a830de93-ddc5-b763-dc1a-35580
    a31401c
VMName                              : IP0051
AssignedImage                       : 57fc60c3-eb9b-4d38-8646-0
    afceec85335
BootedImage                         : 57fc60c3-eb9b-4d38-8646-0
    afceec85335
HostingUnitUid                      : ea17840f-cf2d-4d80-94e0-3
    b752b32e0af
HypervisorConnectionUid             : 99f9f826-31fc-4453-8ca0-9
    ba54306c3ac
IdentityDiskIndex                   : 1
LastBootTime                        : 17/05/2021 09:35:22
OSDiskIndex                         : 0
PersonalVDiskIndex                  : -2147483648
PersonalVDiskStorage                :
StorageId                           : 33ad07a7-edd7-589b-716a-86
    cad4739f5e
NetworkMaps                         : {
 0 }

IdentityDiskId                      : MCS-IdentityDisk-0001
IdentityDiskStorage                 : {
 XDHyp:\Connections\XDHost\Local-Storage-For-Identity-Disk.Storage }

PersonalVDiskId                     :
SecurityGroups                      : {
  }

AdditionalStorage                   : {
  }

WriteBackCacheMemorySize            : 0
UseFullDiskCloneProvisioning        : False
CustomVmData                        :
WorkgroupMachine                    : False
WindowsActivationStatus             : Errored
WindowsActivationTypeProvisionedVirtualMachine : MultipleActivationKey
WindowsActivationStatusErrorCode    : KeyLimitExceeded
WindowsActivationStatusError        : Windows activation failed because
    the maximum number of activations have been exceeded for the
    specified activation key
```

EXAMPLE 2

Gets all the Virtual Machines that were locked, regardless of which Provisioning Scheme the VM was created with.

```
Get-ProvVM -Locked $true

ADAccountName                       : MYDomain\computer1$
```



```
ADAccountSid                          : S
    -1-5-21-3751941309-1176885247-1409628468-3178
CpuCount                              : 1
CreationDate                          : 17/05/2021 09:35:30
Domain                                : steve.dum.local
ImageOutOfDate                        : False
Lock                                  : True
MemoryMB                              : 512
ProvisioningSchemeName                : XenPS
ProvisioningSchemeUid                 : 5135a865-ba49-4e5f-87f2-2
    d65ee7a4e51
ProvisioningSchemeUpdateRequested     : 17/05/2021 09:39:03
ProvisioningSchemeVersion             : 1
Tag                                   : Brokered
VMId                                  : a830de93-ddc5-b763-dc1a-35580
    a31401c
VMName                                : IP0051
AssignedImage                         : 57fc60c3-eb9b-4d38-8646-0
    afceec85335
BootedImage                           : 57fc60c3-eb9b-4d38-8646-0
    afceec85335
HostingUnitUid                        : ea17840f-cf2d-4d80-94e0-3
    b752b32e0af
HypervisorConnectionUid               : 99f9f826-31fc-4453-8ca0-9
    ba54306c3ac
IdentityDiskIndex                     : 1
LastBootTime                          : 17/05/2021 09:35:22
OSDiskIndex                           : 0
PersonalVDiskIndex                    : -2147483648
PersonalVDiskStorage                  :
StorageId                             : 33ad07a7-edd7-589b-716a-86
    cad4739f5e
NetworkMaps                           : {
 0 }

IdentityDiskId                        : MCS-IdentityDisk-0001
IdentityDiskStorage                   : {
 XDHyp:\Connections\XDHost\Local-Storage-For-Identity-Disk.Storage }

PersonalVDiskId                       :
SecurityGroups                        : {
  }

AdditionalStorage                     : {
  }

WriteBackCacheMemorySize              : 0
UseFullDiskCloneProvisioning          : False
CustomVmData                          :
WorkgroupMachine                      : False
WindowsActivationStatus               : Errored
WindowsActivationTypeProvisionedVirtualMachine : MultipleActivationKey
WindowsActivationStatusErrorCode      : KeyLimitExceeded
```



```
49 WindowsActivationStatusError      : Windows activation failed because
       the maximum number of activations have been exceeded for the
       specified activation key
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-VMName

The name of the VM in the hypervisor.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Locked

Indicates whether only VMs that are marked as locked are returned or not (see Lock-ProvVM and Unlock-ProvVM for details).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Tag

The tag string that was associated with the VM when it was locked.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-OutOfDate

Indicates that the image currently assigned to the VM is out of date. The image will be updated the next time the VM is restarted.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

The object has the following properties:

ADAccountSid <string>

The SID of the AD computer account that the VM is using.

ADAccountName <string>

The name of the AD computer account that the VM is using.

CpuCount <int>

The number of processors allocated to the VM.

CreationDate <DateTime>

The date and time when the VM was created.

Domain <string>

The Domain of the AD computer account that the VM is using.

ImageOutOfDate <bool>

Indicates if the image will be updated next time the VM is started.

Lock <bool>

Indicates whether the VM is locked or not.

MemoryMB <int>

The maximum amount of memory allocated to the VM.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the VM.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the VM.

PropertyUpdateWindowStart <DateTime>

Defines the start of the time window wherein if the VM is booted, it will be updated to the latest provisioning scheme and personalization settings.

PropertyUpdateWindowEnd <DateTime>

Defines the end of the time window wherein if the VM is booted, it will be updated to the latest provisioning scheme and personalization settings.

ProvisioningSchemeVersion <int>

The version of the provisioning scheme currently used by this VM.

ProvVMConfigurationVersion <int>

The version of the custom configuration currently used by this VM.

Tag <string>

Provides the string associated with a locked VM.

VMId <string>

The identifier for the VM in the hypervisor.

VMName <string>

The name of the VM in the hypervisor.

AssignedImage <string>

The identifier (in the hypervisor) for the hard disk image that the VM is currently assigned.

BootedImage <string>

The identifier (in the hypervisor) for the hard disk image with which the VM is currently started.

HostingUnitUid <Guid>

The unique identifier of the hosting unit that was used to create the VM.

HypervisorConnectionUid <Guid>

The unique identifier of the hypervisor connection that was used to create the VM.

IdentityDiskIndex <int>

The disk index on which the identity disk is attached.

LastBootTime <DateTime>

The date and time of the last start of the VM.

OSDiskIndex <int>

The disk index on which the hard disk image, from which the VM is currently started, is attached. It is set to [int]::MinValue for VMs inherited from versions of XenDesktop 5.6 or earlier.

PersonalVDiskIndex <int>

The disk index on which the personal vdisk is attached. Defaults to [int]::MinValue for VMs without a personal vDisk.

PersonalVDiskStorage <string>

The identifier (in the hypervisor) for the storage on which the personal vDisk image attached to the VM is located. This is set only if the VM has a personal vDisk attached.

StorageId <string>

The identifier (in the hypervisor) for the storage on which the hard disk image, from which the VM is currently started, is located.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings Used to create the VM.

IdentityDiskId <string>

The identifier (in the hypervisor) for the identity disk attached to the VM.

IdentityDiskStorage <string>

The identifier (in the hypervisor) for the storage on which the identity disk image from which the VM is currently started is located.

PersonalVDiskId <string>

The identifier (in the hypervisor) for the personal vDisk attached to the VM. This is set only if the VM has a personal vDisk attached.

SecurityGroups <string[]>

The security groups that has been applied to the VM. This only applies in a Cloud provisioning context.

AdditionalStorage <System.Collections.Generic.Dictionary[string, AdditionalStorageDiskImage]>

Additional storage disk images attached to this VM keyed by their storage tier names.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache if specified in MB.

UseFullDiskCloneProvisioning <bool>

Indicates if the VM was created using the dedicated full disk clone feature.

CustomVmData <string>

Hypervisor specific custom data packet that has been associated with the virtual machine if any.

WorkgroupMachine <bool>

Indicates if the VM is joined to a workgroup.

WindowsActivationType <string>

Windows Activation Type of the Provisioned Virtual Machine. The default value is unknown and the field is populated once the virtual machine registers with DDC. This is supported only on 2303 and successive versions.

WindowsActivationStatus <string>

Windows Activation Status of the provisioned virtual machine. This is by default set to unknown and is updated when the vda registers with the DDC. The updated values can be Active:The virtual machine is licensed and activated, Inactive:The virtual machine is unlicensed, Errored:An error occured while trying to activate the system.

WindowsActivationStatusErrorCode <string>

The default value is Unknown. If successful, it displays as Success, else it displays with the associated error code.

WindowsActivationStatusVirtualMachineError <string>

Provides a detailed information of potential error message seen while activating the windows system

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData

Only a subset of the available data was returned.

CouldNotQueryDatabase

The query required to get the database was not defined.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

CommunicationError

An error occurred while communicating with the service.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvVM
- Remove-ProvVM
- Lock-ProvVM
- Unlock-ProvVM

Get-ProvVMConfiguration

March 11, 2024

Gets the custom configuration properties for virtual machines created using Machine Creation Services.

Syntax

```
Get-ProvVMConfiguration
    [-ProvisioningSchemeUid <Guid>]
    [[-ProvisioningSchemeName] <String>]
    [-VMName <String>]
    [-VMId <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to list current configuration properties for virtual machines created using Machine Creation Services.

The returned list of properties details what has been applied to VMs specifically with Set-ProvVM. It does not consistute the full set of properties VMs will have when the configuration is applied, to view that information use Get-ProvVMConfigurationResultantSet

Examples

EXAMPLE 1

Configuration is listed on the ProvScheme. Get-ProvVMConfiguration is used to obtain configuration data for all machines in test-catalog. Compare with Example 1 under Get-ProvVMConfigurationResultantSet

```
Get-ProvScheme -ProvisioningSchemeName test-catalog | select
    ProvisioningSchemeName, ServiceOffering, CustomProperties
ProvisioningSchemeName : test-catalog
ServiceOffering        : serviceoffering.folder\Standard_D2_v3.
    serviceoffering
```



```
CustomProperties       : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                           <Property xsi:type="StringProperty" Name="
                              SchemaVersion" Value="2" />
                           <Property xsi:type="StringProperty" Name="
                              UseManagedDisks" Value="true" />
                           <Property xsi:type="StringProperty" Name="
                              OsType" Value="Windows" />
                           </CustomProperties>
...

Get-ProvVMConfiguration -ProvisioningSchemeName test-catalog

CpuCount               :
CustomProperties       : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                           <Property xsi:type="StringProperty" Name="
                              SchemaVersion" Value="2" />
                           <Property xsi:type="StringProperty" Name="
                              LicenseType" Value="Windows_Client" />
                           </CustomProperties>
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : test-catalog
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        :
VMId                   : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata             :
VMName                 : machine1
Version                : 2

CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : test-catalog
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        : serviceoffering.folder\Standard_D2s_v5.
    serviceoffering
VMId                   : d04f4677-f4e7-4c92-ae44-178c22545395
VMMetadata             :
VMName                 : machine2
Version                : 1
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-VMName

The name of the VM in the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-VMId

The ID of the VM in the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisionedMachineConfiguration

The object has the following properties:

Version <int>

The current version of the configuration

CpuCount <int>

The number of processors allocated to the VM.

MemoryInMB <int>

The maximum amount of memory allocated to the VM.

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in cloud hypervisors.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the VM.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the VM.

VMId <string>

The identifier for the VM in the hypervisor.

VMName <string>

The name of the VM in the hypervisor.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Set-ProvVM
- Get-ProvVMConfigurationResultantSet

Get-ProvVMConfigurationResultantSet

March 11, 2024

Gets the resultant configuration properties for virtual machines created using Machine Creation Services.

Syntax

```
Get-ProvVMConfigurationResultantSet
    [-ProvisioningSchemeUid <Guid>]
    [[-ProvisioningSchemeName] <String>]
    [-VMName <String>]
    [-VMId <String>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to list the resultant configuration properties for virtual machines created using Machine Creation Services. This merges properties at the provisioning scheme level with those set on a machine with Set-ProvVM specifically.

To configure a machine with custom configuration data, see Set-ProvVM To list just the properties specific to a machine, use Get-ProvVMConfiguration

Examples

EXAMPLE 1

Configuration is listed on the ProvScheme. Get-ProvVMConfigurationResultantSet is used to obtain configuration data for all machines in test-catalog. Compare with Example 1 under Get-ProvVMConfiguration

```
Get-ProvScheme -ProvisioningSchemeName test-catalog | select
    ProvisioningSchemeName, ServiceOffering, CustomProperties
ProvisioningSchemeName : test-catalog
ServiceOffering        : serviceoffering.folder\Standard_D2_v3.
    serviceoffering
CustomProperties       : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                           <Property xsi:type="StringProperty" Name="
                              SchemaVersion" Value="2" />
                           <Property xsi:type="StringProperty" Name="
                              UseManagedDisks" Value="true" />
                           <Property xsi:type="StringProperty" Name="
                              OsType" Value="Windows" />
```



```
                          </CustomProperties>
...

Get-ProvVMConfigurationResultantSet -ProvisioningSchemeName test-
    catalog

CpuCount                : 
CustomProperties        : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                          <Property xsi:type="StringProperty" Name="
                             SchemaVersion" Value="2" />
                          <Property xsi:type="StringProperty" Name="
                             UseManagedDisks" Value="true" />
                          <Property xsi:type="StringProperty" Name="
                             OsType" Value="Windows" />
                          <Property xsi:type="StringProperty" Name="
                             LicenseType" Value="Windows_Client" />
                          </CustomProperties>
MachineProfile          : 
MemoryInMB              : 
ProvisioningSchemeName  : test-catalog
ProvisioningSchemeUid   : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering         : serviceoffering.folder\Standard_D2_v3.
    serviceoffering
VMId                    : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata              : 
VMName                  : machine1
Version                 : 2

CpuCount                : 
CustomProperties        : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                          <Property xsi:type="StringProperty" Name="
                             SchemaVersion" Value="2" />
                          <Property xsi:type="StringProperty" Name="
                             UseManagedDisks" Value="true" />
                          <Property xsi:type="StringProperty" Name="
                             OsType" Value="Windows" />
                          </CustomProperties>
MachineProfile          : 
MemoryInMB              : 
ProvisioningSchemeName  : test-catalog
ProvisioningSchemeUid   : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering         : serviceoffering.folder\Standard_D2s_v5.
    serviceoffering
VMId                    : d04f4677-f4e7-4c92-ae44-178c22545395
VMMetadata              : 
VMName                  : machine2
Version                 : 1
```


Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-VMName

The name of the VM in the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	True

-VMId

The ID of the VM in the hypervisor.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

See about_Prov_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

See about_Prov_Filtering for details.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

See about_Prov_Filtering for details.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisionedMachineConfigurationResultantSet

The object has the following properties:

CpuCount <int>

The number of processors allocated to the VM.

MemoryInMB <int>

The maximum amount of memory allocated to the VM.

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the VM.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the VM.

VMId <string>

The identifier for the VM in the hypervisor.

VMName <string>

The name of the VM in the hypervisor.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvVM
- Get-ProvVM
- Get-ProvVMConfigurationResultantSet

Import-ProvScheme

March 11, 2024

Import a provisioning scheme into the site

Syntax

```
Import-ProvScheme
      [-ProvisioningSchemeData <String>]
      [-Scope <String[]>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet imports a provisioning scheme into the site. The data passed to this cmdlet should generally be obtained through the Export-ProvScheme cmdlet from a different site.

Examples

EXAMPLE 1

Import the JSON encoded data that is in $provisioningSchemeData.

```
Import-ProvScheme -ProvisioningSchemeData $provisioningSchemeData

CleanOnBoot                  : True
ControllerAddress            : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }

CpuCount                     : 1
DiskSize                     : 20
HostingUnitName              : HostUnit1
HostingUnitUid               : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName             : idPool1
IdentityPoolUid              : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                 : 0
MachineProfile               :
MasterImageVM                : Base.vm/base.snapshot
MasterImageVMDate            : 17/05/2020 09:53:40
MemoryMB                     : 1024
Metadata                     : {
 Department = Sales }
```



```
20
21 MetadataMap                   : {
22  [Department = Sales] }
23
24 ProvisioningSchemeName        : Scheme2
25 ProvisioningSchemeUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
26 ProvisioningSchemeVersion     : 1
27 TaskId                        : 00000000-0000-0000-0000-000000000000
28 VMMetadata                    : {
29  0, 1, 0, 0... }
30
31 PersonalVDiskDriveLetter      :
32 PersonalVDiskDriveSize        : 0
33 UsePersonalVDiskStorage       : False
34 NetworkMaps                   : {
35  0 }
36
37 Scopes                        :
38 DedicatedTenancy              : False
39 GpuTypeId                     :
40 ResetAdministratorPasswords   : False
41 SecurityGroups                : {
42   }
43
44 ServiceOffering               :
45 TenancyType                   : Shared
46 AzureAdJoinType               :
47 CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
48 CustomProperties              :
49 IdentityType:                 : ActiveDirectory
50 UseFullDiskCloneProvisioning : False
51 UseWriteBackCache             : True
52 WriteBackCacheDiskSize        : 24
53 WriteBackCacheMemorySize      : 256
54 Warnings                      : {
55   }
56
57 WriteBackCacheDiskIndex       : 0
```

Parameters

-ProvisioningSchemeData

JSON encoded string of a provisioning scheme, image history, and VMs created with the provisioning scheme.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-Scope

The administration scopes to be applied to the new provisioning scheme.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe a JSON encoded string of a provisioning scheme to Import-ProvScheme.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

CleanOnBoot <bool>

Indicates whether the VMs created will be reset to a clean state on each start.

ControllerAddress <string[]>

The DNS names of the controllers associated with this provisioning scheme.

CpuCount <int>

The number of processors that will be used to create VMs.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

HostingUnitName <string>

The name of the hosting unit being used by this provisioning scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this provisioning scheme.

IdentityPoolName <string>

The name of the identity pool being used by this provisioning scheme.

IdentityPoolUid <Guid>

The unique identifier of the identity pool being used by this provisioning scheme.

MachineCount <int>

The count of machines created with this provisioning scheme.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

MasterImageVM <string>

The inventory path to the VM snapshot copy used by this provisioning scheme.

MasterImageVMDate <DateTime>

The date and time when the VM snapshot copy used by this provisioning scheme was made.

MemoryMB <int>

The maximum amount of memory that will be used to created VMs in MB.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The metadata associated with this provisioning scheme.

MetadataMap <IDictionary[string, string];>

The metadata associated with this provisioning scheme arranged in key value pairs.

ProvisioningSchemeName <string>

The name of this provisioning scheme.

ProvisioningSchemeUid <Guid>

The unique identifier for this provisioning scheme.

ProvisioningSchemeVersion <int>

The version of the provisioning scheme.

TaskId <Guid>

The identifier of any current task that is running for the provisioning scheme.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

PersonalVDiskDriveLetter <char>

The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int>

The size of the personal vDisk in GB.

UsePersonalVDiskStorage <bool>

Indicates whether this provisioning scheme uses personal vDisk storage.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

Scope <Citrix.MachineCreation.Sdk.ScopeReference[]>

The administration scopes associated with this provisioning scheme.

DedicatedTenancy <bool>

Indicates whether dedicated tenancy is used when creating VMs in Cloud Hypervisors.

GpuTypeId <string>

The id of the GPU (Graphics Processor Unit) Type used by this scheme. It is null if there is no GPU.

ResetAdministratorPasswords <bool>

Indicates whether the passwords for administrator accounts are reset on created machines.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

TenancyType <string>

Tenancy type to be used when creating VMs in Cloud Hypervisors. (See New-ProvScheme.)

AzureAdJoinType <string>

Deprecated.

CurrentMasterImageUid <Guid>

The unique identifier of the current master image used by the provisioning scheme. (See Get-ProvSchemeMasterVMImageHistory.)

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

IdentityType <string>

Identity type used to join created machines to a directory service. (See New-ProvScheme.)

UseFullDiskCloneProvisioning <bool>

Indicates whether VMs will be created using the dedicated full disk clone feature.

UseWriteBackCache <bool>

Indicates whether this provisioning scheme will use the write-back cache feature.

WriteBackCacheDiskSize <int>

The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int>

The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

Notes

In the case of failure, the following errors can result.

Error Codes

CouldNotQueryDatabase The query required to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

Related Links

- about_ProvMachineCreationSnapin
- Export-ProvScheme
- Export-AcctIdentityPool
- Import-AcctIdentityPool

Import-ProvVM

March 11, 2024

Import provisioning virtual machines under the given provisioning scheme.

Syntax

```
Import-ProvVM
       [-ProvisioningSchemeName <String>]
       -ProvisioningVirtualMachineData <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Import-ProvVM
       [-ProvisioningSchemeUid <Guid>]
       -ProvisioningVirtualMachineData <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This cmdlet reads the list of provisioning virtual machines (VMs) of a given provisioning scheme and imports them to the database. The list of VMs can be obtained from the output of the Export-ProvScheme cmdlet. If a VM in the list already exists in the database, it will fail silently. If the provisioning scheme the VM is associated with does not exist, it will return an error. This cmdlet returns a list of VM objects imported successfully.

Examples

EXAMPLE 1

Import the JSON encoded provisioning virtual machines that is in $ExportedProvVMs

```
Import-ProvVM -ProvisioningSchemeName ExampleCatalog -
    provisioningVirtualMachineData $ExportedProvVMs

FailureReason                                : The VM already exists
ImportStatus                                 : Failed
WindowsActivationStatusVirtualMachineError   : The information is
    currently unavailable
```



```
ADAccountName                                        :
ADAccountSid                                         : S
    -1-5-21-1369695582-2951410066-511316603-44756
CpuCount                                             : 1
CreationDate                                         : 7/21/2023 11:47:32 AM
Domain                                               : serenity.local
ImageOutOfDate                                       : False
Lock                                                 : False
MemoryMB                                             : 256
PVSDeviceId                                          :
PropertyUpdateWindowEnd                              :
PropertyUpdateWindowStart                            :
ProvVMConfigurationVersion                           :
ProvisioningSchemeName                               : ExampleCatalog
ProvisioningSchemeUid                                : bd9b4150-1d02-47b4-82
    ba-017f7d6e8a19
ProvisioningSchemeVersion                            : 1
Tag                                                  :
VMId                                                 : bd475e5c-f296-54ea-59
    a4-9cff1b9269aa
VMName                                               : HaanVM128
WindowsActivationStatus                              : Unknown
WindowsActivationStatusErrorCode                     : Unknown
WindowsActivationTypeProvisionedVirtualMachine       : UnsupportedVDA
AssignedImage                                        : bda00c96-6e3b-44f0-931
    a-7afeb1ed5371
BootedImage                                          : bda00c96-6e3b-44f0-931
    a-7afeb1ed5371
HostingUnitUid                                       : ec3f578e-d7b2-4a83-
    b3c9-d334798e1448
HypervisorConnectionUid                              : ec3f578e-d7b2-4a83-
    b3c9-d334798e1448
IdentityDiskIndex                                    : 1
LastBootTime                                         : 7/21/2023 11:47:32 AM
OSDiskIndex                                          : 0
PersonalVDiskIndex                                   : -1
PersonalVDiskStorage                                 :
StorageId                                            : 83d77270-8116-1b72-71
    b8-b4b4b5e4e1b2
NetworkMaps                                          : {
  }

IdentityDiskId                                       : 3b37074f-4fdc-4de5-9
    bff-374dfbbc2639
IdentityDiskStorage                                  : 83d77270-8116-1b72-71
    b8-b4b4b5e4e1b2
PersonalVDiskId                                      :
SecurityGroups                                       : {
  }

AdditionalStorage                                    : {
  }

```



```
UseFullDiskCloneProvisioning                    : False
WriteBackCacheMemorySize                        : 0
CustomVmData                                    :
WorkgroupMachine                                : False

FailureReason                                   :
ImportStatus                                    : Success
WindowsActivationStatusVirtualMachineError      : The information is
    currently unavailable
ADAccountName                                   :
ADAccountSid                                    : S
    -1-5-21-1369695582-2951410066-511316603-44761
CpuCount                                        : 1
CreationDate                                    : 7/21/2023 11:47:32 AM
Domain                                          : serenity.local
ImageOutOfDate                                  : False
Lock                                            : False
MemoryMB                                        : 256
PVSDeviceId                                     :
PropertyUpdateWindowEnd                         :
PropertyUpdateWindowStart                       :
ProvVMConfigurationVersion                      :
ProvisioningSchemeName                          : ExampleCatalog
ProvisioningSchemeUid                           : bd9b4150-1d02-47b4-82
    ba-017f7d6e8a19
ProvisioningSchemeVersion                       : 1
Tag                                             :
VMId                                            : fbe4a32a-adb6-52a8
    -6850-974e67a0eb3f
VMName                                          : HaanVM129
WindowsActivationStatus                         : Unknown
WindowsActivationStatusErrorCode                : Unknown
WindowsActivationTypeProvisionedVirtualMachine  : UnsupportedVDA
AssignedImage                                   : bda00c96-6e3b-44f0-931
    a-7afeb1ed5371
BootedImage                                     : bda00c96-6e3b-44f0-931
    a-7afeb1ed5371
HostingUnitUid                                  : ec3f578e-d7b2-4a83-
    b3c9-d334798e1448
HypervisorConnectionUid                         : ec3f578e-d7b2-4a83-
    b3c9-d334798e1448
IdentityDiskIndex                               : 1
LastBootTime                                    : 7/21/2023 11:47:32 AM
OSDiskIndex                                     : 0
PersonalVDiskIndex                              : -1
PersonalVDiskStorage                            :
StorageId                                       : 83d77270-8116-1b72-71
    b8-b4b4b5e4e1b2
NetworkMaps                                     : {
  }

IdentityDiskId                                  : f18788ac-053d-4d9d
    -9093-181a2bc66d06
```



```
IdentityDiskStorage                          : 83d77270-8116-1b72-71
   b8-b4b4b5e4e1b2
PersonalVDiskId                              :
SecurityGroups                               : {
  }

AdditionalStorage                            : {
  }

UseFullDiskCloneProvisioning                 : False
WriteBackCacheMemorySize                     : 0
CustomVmData                                 :
WorkgroupMachine                             : False
```

Parameters

-ProvisioningVirtualMachineData

JSON encoded string of VMs created with the provisioning scheme

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImportProvVmModel

The returned objects provides details of the provisioning virtuam machines and contains the following information.

FailureReason <string> The reason for the failure of importing the VM.

ImportStatus <string> The status of the imported VM.

ADAccountName <string> The name of the AD computer account that the VM is using.

ADAccountSid <string> The SID of the AD computer account that the VM is using.

CpuCount <int> The number of processors allocated to the VM.

CreationDate <DateTime> The date and time when the VM was created.

Domain <string> The Domain of the AD computer account that the VM is using.

ImageOutOfDate <bool> Indicates if the image will be updated next time the VM is started.

Lock <bool> Indicates whether the VM is locked or not.

MemoryMB <int> The maximum amount of memory allocated to the VM.

PropertyUpdateWindowEnd <DateTime> Defines the end of the time window wherein if the VM is booted, it will be updated to the latest provisioning scheme and personalization settings.

PropertyUpdateWindowStart <DateTime> Defines the start of the time window wherein if the VM is booted, it will be updated to the latest provisioning scheme and personalization settings.

ProvisioningSchemeName <string> The name of the provisioning scheme associated with the VM.

ProvisioningSchemeUid <Guid> The unique identifier of the provisioning scheme associated with the VM.

ProvisioningSchemeVersion <int> The version of the provisioning scheme currently used by this VM.

ProvVMConfigurationVersion <int> The version of the custom configuration currently used by this VM.

Tag <string> Provides the string associated with a locked VM.

VMId <string> The identifier for the VM in the hypervisor.

VMName <string> The name of the VM in the hypervisor.

AssignedImage <string> The identifier (in the hypervisor) for the hard disk image that the VM is currently assigned.

BootedImage <string> The identifier (in the hypervisor) for the hard disk image with which the VM is currently started.

HostingUnitUid <Guid> The unique identifier of the hosting unit that was used to create the VM.

HypervisorConnectionUid <Guid> The unique identifier of the hypervisor connection that was used to create the VM.

IdentityDiskIndex <int> The disk index on which the identity disk is attached.

LastBootTime <DateTime> The date and time of the last start of the VM.

OSDiskIndex <int> The disk index on which the hard disk image, from which the VM is currently started, is attached. It is set to [int]::MinValue for VMs inherited from versions of XenDesktop 5.6 or earlier.

PersonalVDiskIndex <int> The disk index on which the personal vdisk is attached. Defaults to [int]::MinValue for VMs without a personal vDisk.

PersonalVDiskStorage <string> The identifier (in the hypervisor) for the storage on which the personal vDisk image attached to the VM is located. This is set only if the VM has a personal vDisk attached.

StorageId <string> The identifier (in the hypervisor) for the storage on which the hard disk image, from which the VM is currently started, is located.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]> The NIC/network mappings Used to create the VM.

IdentityDiskId <string> The identifier (in the hypervisor) for the identity disk attached to the VM.

IdentityDiskStorage <string> The identifier (in the hypervisor) for the storage on which the identity disk image from which the VM is currently started is located.

PersonalVDiskId <string> The identifier (in the hypervisor) for the personal vDisk attached to the VM. This is set only if the VM has a personal vDisk attached.

SecurityGroups <string[]> The security groups that has been applied to the VM. This only applies in a Cloud provisioning context.

AdditionalStorage <System.Collections.Generic.Dictionary[string, AdditionalStorageDiskImage]> Additional storage disk images attached to this VM keyed by their storage tier names.

UseFullDiskCloneProvisioning <bool> Indicates if the VM was created using the dedicated full disk clone feature.

WriteBackCacheMemorySize <int> The size of the write-back memory cache if specified in MB.

CustomVmData <string> Hypervisor specific custom data packet that has been associated with the virtual machine if any.

WorkgroupMachine <bool> Indicates if the VM is joined to a workgroup.

WindowsActivationType <string> Windows Activation Type of the Provisioned Virtual Machine. The default value is unknown and the field is populated once the virtual machine registers with DDC. This is supported only on 2303 and successive versions.

WindowsActivationStatus <string> Windows Activation Status of the provisioned virtual machine. This is by default set to unknown and is updated when the vda registers with the DDC. The updated values can be Active:The virtual machine is licensed and activated, Inactive:The virtual machine is unlicensed, Errored:An error occured while trying to activate the system.

WindowsActivationStatusErrorCode <string> The default value is Unknown. If successful, it displays as Success, else it displays with the associated error code.

WindowsActivationStatusVirtualMachineError <string> Provides a detailed information of potential error message seen while activating the windows system

Notes

The scope of Import-ProvVm is limited to importing (newly added) VMs only. It assumes

the items below.

1. Import-AcctIdentityPool is done.
2. Import-ProvScheme is done.
3. The input data to Import-ProvVM includes the list of Provisioning VMs.

Related Links

- Export-ProvScheme
- Import-ProvScheme
- Export-AcctIdentityPool
- Import-AcctIdentityPool

Lock-ProvVM

March 11, 2024

Locks one or more VM(s).

Syntax

```
Lock-ProvVM
    -ProvisioningSchemeName <String>
    [-VMID] <String[]>
    [-Tag <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Lock-ProvVM
    -ProvisioningSchemeUid <Guid>
    [-VMID] <String[]>
    [-Tag <String>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to ‘lock’virtual machine(s) with a tag string. This indicates that the VM is being used, and prevents other commands from removing the VM without unlocking it.

Examples

EXAMPLE 1

Locks the VM with the Id ‘bc79802c-ba6e-8de8-99e9-4c35d7ad24b4’in the provisioning scheme ‘MyScheme’with the tag ‘LockedString’.

```
Lock-ProvVM -provisioningSchemeName MyScheme -VMId  bc79802c-ba6e-8de8
    -99e9-4c35d7ad24b4 -Tag LockedString
```

EXAMPLE 2

Locks all the VMs in the provisioning scheme ‘MyScheme’with the tag ‘LockedString’.

```
Get-ProvVM -provisioningSchemeName MyScheme | Lock-ProvVM -Tag
    LockedString
```


Parameters

-VMID

Array of the virtual machine Ids (hypervisor context).

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeName

The name of the provisioning scheme for which VMs must be locked.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme for which VMs must be locked.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	True

-Tag

The string to be held against the VM being locked.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId'and 'ProvisioningSchemeName'to Lock-ProvVM.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

VMDoesNotExist

The specified VM cannot be located.

VMAlreadyLocked

The VM is already unlocked.

VMDoesNotExistForProvisioningScheme

The specified VM does exist in the hypervisor, but is not part of the specified provisioning scheme.

PermissionDenied

The user is not authorized to perform this operation

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- UnLock-ProvVM
- Get-ProvVM
- Remove-ProvVM

Move-ProvVMDisk

March 11, 2024

Move the disks of full clone virtual machines to a new storage.

Syntax

```
Move-ProvVMDisk
    [-ProvisioningSchemeName] <String>
    -VMName <String[]>
    -DestinationStorageId <String>
    [-DiskType <DiskMovingType>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Move-ProvVMDisk
    [-ProvisioningSchemeUid] <Guid>
    -VMName <String[]>
    -DestinationStorageId <String>
    [-DiskType <DiskMovingType>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to move the disks of full clone virtual machines to a new storage and the new storage info will be updated to MCS database.

Examples

EXAMPLE 1

Move the disks of machine01 in provisioning scheme myProvScheme to storage datastore1

```
Move-ProvVMDisk -ProvisioningSchemeName "myProvScheme" -VMName "
    machine01" -DestinationStorageId datastore1
```

EXAMPLE 2

Move the disks of machine01 and machine02 in provisioning scheme myProvScheme to storage datastore1

```
Move-ProvVMDisk -ProvisioningSchemeName "myProvScheme" -VMName ("
    machine01","machine02") -DestinationStorageId datastore1
```


EXAMPLE 3

Move all VMs'disks in provisioning scheme myProvScheme to storage datastore1

```
1  ,(Get-ProvVM -ProvisioningSchemeName "myProvScheme") | Move-ProvVMDisk
      -ProvisioningSchemeName "myProvScheme" -DestinationStorageId
      datastore1
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme from which VMs'disks will be moved.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme from which VMs'disks will be moved.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMName

List of VM names that will be have the disk moving.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-DestinationStorageId

Indicates the destination storageId. The details can refer to Get-ProvSchemeResourceInStorage

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DiskType

Indicates the type of disks which will be moved

Type:	DiskMovingType
Position:	Named
Default value:	All
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether an identifier for the task must returned before completion of the task. The progress of the task can be monitored using 'get-ProvTask'command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMName'and 'ProvisioningSchemeName'to move disks.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored for the task. For new tasks, this is empty until metadata is added.

CurrentOperation <string>

Operation specific phase of the overall “Running”task state.

TaskProgress <double>

The progress of the task 0-100%.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For move vm disks task, this is always “Move-ProvVMDisk”.

Status <string>

Where in its lifecycle the task is.

DiskMovedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

See DiskMoveFailedVirtualMachines for details of the object parameters.

DiskMoveFailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

ADAccountName <string>

The domain-qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents.

Message <string>

The exception message.

Details <string>

The full exception content including stack trace.

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state.

Status <string>

The status of the virtual machine.

StatusAdditionalInformation <string>

Extra information about the Status.

VMId <string>

The virtual machine identifier in the hypervisor.

VMName <string>

The display name of the virtual machine in the hypervisor.

NotStartedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

See FailedVirtualMachines for details of the object parameters.

ProvisioningSchemeName <string>

The name of the provisioning scheme from which the VM was created.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme from which the VM was created.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,

Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

Job failed for reasons specified in TaskStateInformation.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>

Provides more detailed information about the current task state.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

Notes

The cmdlet is associated with a task of type MoveVMDiskTask, and while active will move through the following operations (CurrentOperation field):

ValidatingInputs

MoveVMDisks

ReplicateBaseDisks

RemoveStaleBaseDsik

Only one long-running task for each provisioning scheme can be processed at a time.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

InvalidParameterCombination

Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Get-ProvSchemeResourceInStorage

New-ProvImageDefinition

March 11, 2024

Create a new image definition.

Syntax

```
New-ProvImageDefinition
   [-PreparedImageDefinitionName] <String>
   [-Description <String>]
   -HostingUnitName <String>
   -OsType <String>
   [-CustomProperties <String>]
   [-UseWriteBackCache]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
New-ProvImageDefinition
   [-PreparedImageDefinitionName] <String>
   [-Description <String>]
   -HostingUnitUid <Guid>
   -OsType <String>
   [-CustomProperties <String>]
   [-UseWriteBackCache]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Lets you create a new image definition. An image definition can contain multiple image versions (See New-ProvImageVersion). The image definition defines basic information of all its image versions.

Examples

EXAMPLE 1

Create a new image definition with name “MyImage”on HostingUnit Hu.

```
New-ProvImageDefinition -PreparedImageDefinitionName MyImage -
   HostingUnitName Hu -OsType Windows -Description "Windows image"

  CreateTime                     : 1/1/2022 12:00:00
  CustomProperties               :
  Description                    : Windows image
  HostingUnitName                : Hu
  HostingUnitUid                 : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
  OsType                         : Windows
  PreparedImageDefinitionName    : MyImage
  PreparedImageDefinitionUid     : 7585f0de-192e-4847-a6d8-22713c3a2f42
  UseWriteBackCache              : False
```

Parameters

-PreparedImageDefinitionName

The name of the image definition to be created.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-OsType

A value indicating the OS type for the image versions in this image definition.

Type:	String
Accepted values:	Linux, Windows
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitName

The name of the hosting unit used for the image definition.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitUid

The identifier for the hosting unit used for the image definition.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

A value indicating the image description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CustomProperties

The properties of the image scheme that are specific to the target hosting infrastructure.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseWriteBackCache

Indicates whether or not write back cache is enabled for the image versions in this image definition.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageDefinition

This object provides details of the image definitions and contains the following information:

CreateTime <DateTime>

The date and time when the image definition was created.

CustomProperties <string>

Properties of the image definition which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

Description <string>

The description for this image definition.

HostingUnitName <string>

The name of the hosting unit being used by this image definition.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this image definition.

OsType <string>

The OS type for this image definition.

PreparedImageDefinitionName <string>

The name of this image Definition.

PreparedImageDefinitionUid <Guid>

The unique identifier for this image definition.

UseWriteBackCache <bool>

Whether to use WriteBackCache of this image definition.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageDefinition
- Remove-ProvImageDefinition
- Rename-ProvImageDefinition
- Set-ProvImageDefinition

New-ProvImageScheme

March 11, 2024

Creates a new image scheme.

Syntax

```
New-ProvImageScheme
   [-PreparedImageSchemeName] <String>
   -HostingUnitName <String>
   [-VMCpuCount <Int32>]
   [-VMMemoryMB <Int32>]
   -NetworkMapping <Hashtable>
   [-ServiceOffering <String>]
   [-CustomProperties <String>]
   [-MachineProfile <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-ProvImageScheme
   [-PreparedImageSchemeName] <String>
   -HostingUnitUid <Guid>
   [-VMCpuCount <Int32>]
   [-VMMemoryMB <Int32>]
   -NetworkMapping <Hashtable>
   [-ServiceOffering <String>]
   [-CustomProperties <String>]
   [-MachineProfile <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Lets you create a new image scheme. The image scheme is a collection of all of the data that is required to prepare a template against which virtual machines can be created. An image scheme will be used in New-ImageVersion command to prepare an image.

It therefore requires the following: Hosting Unit A reference to an item defined in the Host Service that defines the hypervisor, the network, and the storage to be used.

This cmdlet requires information to be provided that is retrieved using other snap-ins that form part of the Citrix Machine Creation Services: Hosting Unit Service Snapin The snap-in that provides information about the hypervisors.

Examples

EXAMPLE 1

Creates a new image scheme with the name “XenPS”using the hosting unit “XenHu”.

```
New-ProvImageScheme -PreparedImageSchemeName XenPS -HostingUnitName
    XenHu

CpuCount                  : 0
CustomProperties          :
HostingUnitName           : XenHu
HostingUnitUid            : e5d4a2c9-6650-4cfc-93cc-1686db82b92e
MachineProfile            :
MemoryMB                  : 0
NetworkMaps               : []
PreparedImageSchemeName   : XenPS
PreparedImageSchemeUid    : 85bd21c6-851b-420a-a702-2fa2ee6f0052
ServiceOffering           :
```

EXAMPLE 2

Creates a new image scheme with the name “azurescheme”using the hosting unit “azure”.

```
New-ProvImageScheme -PreparedImageSchemeName azurescheme -
    HostingUnitName azure -ServiceOffering "XDHyp:\HostingUnits\azure\
    serviceoffering.folder\Standard_B2ms.serviceoffering" -
    NetworkMapping @{
 "0"="XDHyp:\HostingUnits\azure\virtualprivatecloud.folder\East US.
     region\virtualprivatecloud.folder\hu.resourcegroup\hu.
     virtualprivatecloud\default.network" }


CpuCount                  : 0
CustomProperties          :
HostingUnitName           : azure
HostingUnitUid            : e5d4a2c9-6650-4cfc-93cc-1686db82b92e
MachineProfile            :
MemoryMB                  : 0
NetworkMaps               : [{
 "DeviceId":"0","NetworkId":"","NetworkPath":"virtualprivatecloud.
     folder\\East US.region\\virtualpr
                           ivatecloud.folder\\hu.resourcegroup\\hu.
                              virtualprivatecloud\\default.network" }
 ]
PreparedImageSchemeName   : myImageScheme
PreparedImageSchemeUid    : b5a55401-ad36-4f53-b0e0-dbb6a5bdf6fd
ServiceOffering           : serviceoffering.folder\Standard_B2ms.
    serviceoffering
```


Parameters

-PreparedImageSchemeName

The name of the image scheme to be created. This must be a name that is not being used by an existing image scheme, and it must not contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitName

The name of the hosting unit used for the image scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitUid

The identifier for the hosting unit used for the image scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkMapping

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, provisioned VM in New-ImageVersion command is created with a single NIC, which is mapped to the default network in the HostingUnit. If this parameter is supplied, machine is created with the number of NICs specified in the map, and each NIC is attached to the specified network.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMCpuCount

The number of processors used by virtual machines created from the image scheme.

Type:	Int32
Position:	Named
Default value:	The number of CPUs assigned to the base image VM snapshot or VM template.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMMemoryMB

The maximum amount of memory used by virtual machines created from the image scheme.

Type:	Int32
Position:	Named
Default value:	The amount of memory assigned to the base image VM snapshot or VM template.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceOffering

The Service Offering to use when creating machines in Cloud Hypervisors.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CustomProperties

The properties of the image scheme that are specific to the target hosting infrastructure.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineProfile

The inventory path to the source that is used as a template. This profile identifies the values for the VMs created by the catalog. This must be a path to an item in the same hosting unit that the hosting unit name or hosting unit UID refers to.

Valid paths are of the format:

XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageScheme

This object provides details of the image scheme and contains the following information:

CpuCount <int>

The number of processors that will be used to create VMs.

CustomProperties <string>

Properties of the image scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

HostingUnitName <string>

The name of the hosting unit being used by this image scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this image scheme.

MachineProfile <string>

The inventory path to the source VM used by the image scheme as a template.

MemoryMB <int>

The maximum amount of memory that will be used to created VMs in MB.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

PreparedImageSchemeName <string>

The name of this image scheme.

PreparedImageSchemeUid <Guid>

The unique identifier for this image scheme.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

Notes

In case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

DatabaseMissingCapabilites

The database supporting the service instance being used has not been upgraded to support the personal vDisk feature.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ScopeNotFound

One or more of the scopes nominated for the new image scheme do not exist.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used, or examine the XenDesktop logs.

ImageSchemeNameAlreadyExists

The specified image scheme name has already existed.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageScheme
- Remove-ProvImageScheme

New-ProvImageVersion

March 11, 2024

Prepare an image version in one image definition which can be used by provisioning scheme to provisioning virtual machines.

Syntax

```
New-ProvImageVersion
   -MasterImageVM <String>
   [-PreparedImageVersionNumber] <String>
   -PreparedImageDefinitionName <String>
   -PreparedImageSchemeName <String>
   [-Description <String>]
   [-NoImagePreparation]
   [-WriteBackCacheDiskSize <Int32>]
   [-WriteBackCacheMemorySize <Int32>]
   [-Metadata <Hashtable>]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-ProvImageVersion
   -MasterImageVM <String>
   [-PreparedImageVersionNumber] <String>
   -PreparedImageDefinitionUid <Guid>
   -PreparedImageSchemeUid <Guid>
   [-Description <String>]
```



```
    [-NoImagePreparation]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-Metadata <Hashtable>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The prepared image version can be assigned to provisioning schemes to provision virtual machines

A snapshot is used rather than a VM, so that the content of the hard disk for the image scheme can be easily determined.

As the snapshot is specified using a path into the Citrix Host Service Powershell Provider, the Citrix Host Service Powershell snap-in must also be loaded for this cmdlet to be used.

Examples

EXAMPLE 1

Prepares a new master hard disk image for the myImage version 1 using the “base.snapshot”hard disk image.

```
New-ProvImageVersion -PreparedImageVersionNumber 1 -
    PreparedImageDefinitionName myImage -PreparedImageSchemeName
    myImageScheme -MasterImageVM XDHyp:\H
  stingUnits\HostUnit1\RhoneCC_baseXP.vm\base.snapshot -
     RunAsynchronously

  Guid
  ----
  0828571f-d6c2-4795-bd47-7e58b6586f34

  c:\PS>get-provtask -TaskId 0828571f-d6c2-4795-bd47-7e58b6586f34
  TaskId                     : 0828571f-d6c2-4795-bd47-7e58b6586f34
  Active                     : True
  Host                       : E70972-42-1
  DateStarted                : 4/2/2022 2:57:21 AM
  Metadata                   : {
  }

  Type                       : NewImageVersion
  Status                     : Running
  CurrentOperation           : PreparingMasterImage
```



```
19   TaskExpectedCompletion     :
20   LastUpdateTime             : 4/2/2022 2:57:42 AM
21   ActiveElapsedTime          : 21
22   DateFinished               :
23   TerminatingError           :
24   Storage                    : {
25   }
26
27   WorkflowStatus             : Running
28   Warnings                   : {
29   }
30
31   ImageStatus                :
32   PreparedImageVersionNumber : 1
33   PreparedImageVersionUid    : b1c7602c-b080-4a63-846b-9d2429d54bd1
34   PreparedImageDefinitionName: myImage
35   PreparedImageDefinitionUid : 42bd09f5-3fac-44a6-8b1c-7c09f1079cc0
36   PreparedImageSchemeName    : myImageScheme
37   PreparedImageSchemeUid     : 53ed79c3-c771-4114-ba05-b999a62a836f
38   HostingUnitUid             : 0be23759-e276-4657-b863-e958a716cb6e
39   CustomProperties           :
40   AdditionalCustomProperties :
41   InitialBatchSizeHint       : 1
42   TaskState                  : ImagePreparationInitialization
43   TaskStateInformation       : ImageDefinitionName:/East US.region/
        image.folder/hu.resourcegroup/vda-snapshot.snapshot, StoragePath:,
         HostingUnitName:hu
44   TaskProgress               :
45   WriteBackCacheDiskSize     : 0
46   WriteBackCacheMemorySize   : 0
47   Scopes                     : ??
48   NetworkMaps                : {
49  0 }
50
51   ServiceOffering            : serviceoffering.folder\Standard_B2ms.
        serviceoffering
52   StatusMessageSubstitutions :
```

Parameters

-PreparedImageVersionNumber

The version number of the image version to be created.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MasterImageVM

The path in the hosting unit provider to the virtual machine snapshot that will be used. This identifies the hard disk to be used and the default values for the memory and processors. This must be a path to a Snapshot item in the same hosting unit that the hosting unit name or hosting unit Uid refers to.

Valid paths are of the format; XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm\<SnapshotName>

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name for the image definition used for the image version.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageSchemeName

The name for the image scheme used for the image version.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionUid

The identifier for the image definition used for the image version.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageSchemeUid

The identifier for the image scheme used for the image version.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

A value indicating the image version description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NoImagePreparation

Gets or sets a flag indicating that Image Preparation should be disabled

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheDiskSize

The size in GB of any temporary storage disk used by the write back cache. Should be used in conjunction with WriteBackCacheMemorySize.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheMemorySize

The size in MB of any write back cache if required. Should be used in conjunction with WriteBackCacheDiskSize. Setting RAM Cache to 0 but specifying Disk Cache effectively disables the RAM Cache. However, there will be some memory still used to allow the I/O Optimization to operate.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

Gets or sets metadata of the image version.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether or not the cmdlet should return before it is complete. If specified, the command returns an identifier for the task that was created. You can monitor this task using the Get-ProvTask cmdlet.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history will be removed from the database when the task has finished. This cannot be specified for tasks that are run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <Boolean>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

Type <Citrix.XDInterServiceTypes.JobType>

The type of the task. For new image version tasks this is always "NewImageVersion".

Status <string>

Where in its lifecycle the task is.

CurrentOperation <string>

Operation specific phase of the overall "Running"task state.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateStarted <DateTime>

The date and time when the task was started.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Storage <Citrix.MachineCreation.Sdk.NewProvSchemeStorage[]>

Storage objects to be used for the new image version task.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warnings associated with the new image version task.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

PreparedImageVersionNumber <string>

The version number of the image versions that the task was for.

PreparedImageVersionUid <Guid>

The unique identifier of the image version that the task was for.

PreparedImageDefinitionName <string>

The name of the image definition for which the image version is created.

PreparedImageDefinitionUid <Guid>

The unique identifier of the image definition that the task was for.

PreparedImageSchemeUid <Guid>

The unique identifier of the image scheme which is used for this image version.

MasterImage <string>

The path (in the hosting unit provider) of the virtual machine snapshot that was used as the master image for the task.

Metadata <System.Collections.Generic.Dictionary[string, string];>

The metadata to apply to the image version, if specified.

CustomProperties <string>

The properties of the image version that are specific to the target hosting infrastructure.

InitialBatchSizeHint <int>

The number of VMs that are expected to be added to the scheme as an initial batch.

HostingUnitUid <Guid>

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new image version will use.

TaskState <Citrix.DesktopUpdateManager.SDK.NewProvisioningSchemeState>

The state of the task. This can be any of the following:

Processing

Indicates that the task has just begun and has not done anything yet.

LocatingResources,

Indicates that the workflow is locating resources from other services.

HostingUnitNotFound

Indicates that the task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound

Indicates that the task failed because the required snapshot could not be located.

ConsolidatingMasterImage

Indicates that the task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage

Indicates that the task is replicating the consolidated master image.

StoringImageVersion

Indicates that the task is storing the image version data to the database.

Finished

Indicates that the task has completed with no errors.

ImageVersionAlreadyExists

Indicates that the task failed because a image version with the same name already exists.

MasterVMImageIsNotPartOfImageDefinitionHostingUnit,

Indicates that the hosting unit that the master image originated from is not the hosting unit that the image definition is defined to use.

MasterVmImageIsNotASnapshot

Indicates that the task failed because the master VM Image path does not refer to a Snapshot item.

InvalidMasterVMConfiguration

The VM snapshot specified as the master had an invalid configuration.

InvalidMasterVMState

The VM snapshot specified as the master is currently in an invalid state.

InsufficientResources

Indicates that the task failed because the hypervisor did not have enough resources to complete the task.

StorageNotFound

Indicates that no associated storage was found in the hosting unit.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>

Holds string data providing more details about the current task state.

TaskProgress

The progress of the task 0-100%.

WriteBackCacheDiskSize <int>

The size of any write-back cache disk (zero if the write-back cache feature was not selected).

WriteBackCacheMemorySize <int>

The size of the write-back cache (zero if the write-back cache feature was not selected).

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The list of NIC to network associations, if specified.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

Notes

Only one long-running task per image version can be processed at any one time.

In the case of failure, the following errors can result.

Error Codes

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

The cmdlet is associated with a task of type NewImageVersion, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

ConsolidatingMasterImage

PreparingMasterImage

ReplicatingMasterImage

CommittingScheme

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageVersion
- Remove-ProvImageVersion
- Set-ProvImageVersion

New-ProvScheme

March 11, 2024

Creates a new provisioning scheme.

Syntax

```
New-ProvScheme
   [-ProvisioningSchemeName] <String>
   [-MasterImageVM <String>]
   -HostingUnitName <String>
   [-VMCpuCount <Int32>]
   [-VMMemoryMB <Int32>]
   -IdentityPoolName <String>
   [-CleanOnBoot]
   [-UseWriteBackCache]
   [-Scope <String[]>]
   [-NoImagePreparation]
   [-NetworkMapping <Hashtable>]
   [-Metadata <Hashtable>]
   [-ServiceOffering <String>]
   [-SecurityGroup <String[]>]
   [-TenancyType <String>]
   [-MachineProfile <String>]
```



```
    [-CustomProperties <String>]
    [-ResetAdministratorPasswords]
    [-FunctionalLevel <String>]
    [-UseFullDiskCloneProvisioning]
    [-Validate]
    [-MasterImageNote <String>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-InitialBatchSizeHint <Int32>]
    [-PVSSite <Guid>]
    [-PVSvDisk <Guid>]
    [-ProvisioningSchemeType <ProvisioningSchemeType>]
    [-ForceCreate]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-ProvScheme
    [-ProvisioningSchemeName] <String>
    [-MasterImageVM <String>]
    -HostingUnitUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CleanOnBoot]
    [-UseWriteBackCache]
    [-Scope <String[]>]
    [-NoImagePreparation]
    [-NetworkMapping <Hashtable>]
    [-Metadata <Hashtable>]
    [-ServiceOffering <String>]
    [-SecurityGroup <String[]>]
    [-TenancyType <String>]
    [-MachineProfile <String>]
    [-CustomProperties <String>]
    [-ResetAdministratorPasswords]
    [-FunctionalLevel <String>]
    [-UseFullDiskCloneProvisioning]
    [-Validate]
    [-MasterImageNote <String>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-InitialBatchSizeHint <Int32>]
    [-PVSSite <Guid>]
    [-PVSvDisk <Guid>]
    [-ProvisioningSchemeType <ProvisioningSchemeType>]
    [-ForceCreate]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-ProvScheme
    [-ProvisioningSchemeName] <String>
```



```
    -PreparedImageVersionUid <Guid>
    -HostingUnitUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CleanOnBoot]
    [-UseWriteBackCache]
    [-Scope <String[]>]
    -NetworkMapping <Hashtable>
    [-Metadata <Hashtable>]
    [-ServiceOffering <String>]
    [-SecurityGroup <String[]>]
    [-TenancyType <String>]
    [-MachineProfile <String>]
    [-CustomProperties <String>]
    [-ResetAdministratorPasswords]
    [-UseFullDiskCloneProvisioning]
    [-Validate]
    [-MasterImageNote <String>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-InitialBatchSizeHint <Int32>]
    [-PVSSite <Guid>]
    [-PVSvDisk <Guid>]
    [-ProvisioningSchemeType <ProvisioningSchemeType>]
    [-ForceCreate]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-ProvScheme
    [-ProvisioningSchemeName] <String>
    -PreparedImageDefinitionName <String>
    -PreparedImageVersionNumber <String>
    -HostingUnitName <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolName <String>
    [-CleanOnBoot]
    [-UseWriteBackCache]
    [-Scope <String[]>]
    -NetworkMapping <Hashtable>
    [-Metadata <Hashtable>]
    [-ServiceOffering <String>]
    [-SecurityGroup <String[]>]
    [-TenancyType <String>]
    [-MachineProfile <String>]
    [-CustomProperties <String>]
    [-ResetAdministratorPasswords]
    [-UseFullDiskCloneProvisioning]
    [-Validate]
    [-MasterImageNote <String>]
    [-RunAsynchronously]
```



```
    [-PurgeJobOnSuccess]
    [-InitialBatchSizeHint <Int32>]
    [-PVSSite <Guid>]
    [-PVSvDisk <Guid>]
    [-ProvisioningSchemeType <ProvisioningSchemeType>]
    [-ForceCreate]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to create a new provisioning scheme. The creation process makes a copy of the hard disk attached to a virtual machine snapshot or VM template and stores it in every storage location that the hosting unit referenced by the provisioning scheme defines. This is a long-running task and typically takes several minutes to complete (depending on the size of the hard disk that is being copied and the number of snapshots that the hard disk consists of).

A snapshot or VM template must be used rather than a VM, so that the content of the hard disk for the provisioning scheme can be easily determined.

Because the snapshot or VM template are specified using a path into the Citrix Host Service PowerShell Provider, the Citrix Host Service PowerShell snap-in must also be loaded to use this cmdlet.

This cmdlet requires information to be provided that is retrieved using other snap-ins that form part of the Citrix Machine Creation Services:

Hosting Unit Service Snapin

The snap-in that provides information about the hypervisors.

AD Identity Service Snapin

The snap-in that provides information about the identity pools.

The provisioning scheme is a collection of all of the data that is required to form a template against which virtual machines can be created. It therefore requires the following:

Hosting Unit

A reference to an item defined in the Host Service that defines the hypervisor, the network, and the storage to be used.

Identity Pool

A reference to the collection of Active Directory accounts that is used for VMs created from the provisioning scheme.

Examples

EXAMPLE 1

Creates a new provisioning scheme with the name “XenPS”using the hosting unit “XenHu”and the identity pool “idPool1”from the master VM snapshot called “Base.snapshot”.

```
New-ProvScheme -ProvisioningSchemeName XenPS -HostingUnitName XenHu -
    IdentityPoolName idPool1 -CleanOnBoot -MasterImageVM XDHyp:\
    HostingUnits\XenHU\Base.vm\Base.snapshot

TaskId                                  : 90e93b9d-a225-4701-ad50-
    fa1546af35ac
Type                                    : NewProvisioningScheme
Status                                  : Finished
CurrentOperation                        :
TaskExpectedCompletion                  :
LastUpdateTime                          : 17/05/2020 08:24:08
ActiveElapsedTime                       : 11
DateStarted                             : 17/05/2020 08:22:22
DateFinished                            : 17/05/2020 08:24:08
TerminatingError                        :
Storage                                 : {
  }

WorkflowStatus                          : Completed
Warnings                                : {
  }

ProvisioningSchemeName                  : XenPS
MasterImage                             : XDHyp:\HostingUnits\XenHU\Base.vm
    \Base.snapshot
IdentityPoolName                        : idPool1
IdentityPoolUid                         : 03743136-e43b-4a87-af74-
    ab71686b3c16
HostingUnitName                         : XenHU
HostingUnitUid                          : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
CustomProperties                        :
InitialBatchSizeHint                    : 0
ProvisioningSchemeUid                   : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                               : Finished
TaskStateInformation                    :
TaskProgress                            : 100
DiskSize                                : 24
MasterImageNote                         :
WriteBackCacheDiskSize                  : 127
WriteBackCacheMemorySize                : 256
WriteBackCacheDriveLetter               : W
Scopes                                  : {
  }
```



```
39
40 NetworkMaps                          : {
41  0 }
42
43 ProvisioningSchemeMetadata          : {
44   }
45
46 ServiceOffering                      :
47 SecurityGroups                       :
48 DedicatedTenancy                     : False
49 ResetAdministratorPasswords          : True
50 StatusMessageSubstitutions           :
51 GpuTypeId                            :
52 UseFullDiskCloneProvisioning         : False
```

EXAMPLE 2

Creates a new provisioning scheme with the name “AWS1”using the hosting unit “aws-test”and the identity pool “idPool1”from the master VM snapshot called “TemplateAmi”. The machine instance “server-vda”is used as a Machine Profile when creating the provisioning scheme so all other parameters are taken from the instance.

```
1 $provScheme = New-ProvScheme -ProvisioningSchemeName AWS1 -
      HostingUnitName aws-test -IdentityPoolName idPool1 -MasterImageVM "
      XDHyp:\HostingUnits\aws-test\TemplateAmi (ami-06927522c36bf4109).
      template" -CleanOnBoot -MachineProfile "XDHyp:\HostingUnits\aws-test
      \us-east-1a.availabilityzone\server-vda (i-0c136ee2a01a3dd60).vm"
```

EXAMPLE 3

Creates a new provisioning scheme with the name “AWS2”using the hosting unit “aws-test”and the identity pool “idPool2”from the master VM snapshot called “TemplateAmi”. The version “1”of the launch template with name “machine-profile-lt”and id “lt-06927522c36bg5698”is used as a Machine Profile when creating the provisioning scheme so all other parameters are taken from this launch template version. If instance type value (ServiceOffering) is empty in the machine profile launch template version and no service offering is provided in cmdlet parameter, the cmdlet will return error.

```
1 $provScheme = New-ProvScheme -ProvisioningSchemeName AWS2 -
      HostingUnitName aws-test -IdentityPoolName idPool2 -MasterImageVM "
      XDHyp:\HostingUnits\aws-test\TemplateAmi (ami-06927522c36bf4109).
      template" -CleanOnBoot -MachineProfile "XDHyp:\HostingUnits\aws-test
      \machine-profile-lt (lt-06927522c36bg5698).launchtemplate\lt
      -06927522c36bg5698 (1).launchtemplateversion"
```


EXAMPLE 4

Creates a new provisioning scheme with the name “XenPS”using the hosting unit “XenHu”and the identity pool “idPool1”from the master VM snapshot called “Base.snapshot”asynchronously. To get the task details, use Get-ProvTask -TaskID <task id>.

```
New-ProvScheme -ProvisioningSchemeName XenPS -HostingUnitName XenHu -
    IdentityPoolName idPool1 -CleanOnBoot -MasterImageVM XDHyp:\
    HostingUnits\XenHU\Base.vm\Base.snapshot -RunAsynchronously

Guid
----
6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

Get-ProvTask -TaskID 6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

TaskId                                : 6dd85fec-96cf-46b1-9cd4-
    d8ba7d06e85b
Type                                  : NewProvisioningScheme
Status                                : Finished
CurrentOperation                      :
TaskExpectedCompletion                :
LastUpdateTime                        : 17/05/2020 08:24:08
ActiveElapsedTime                     : 11
DateStarted                           : 17/05/2020 08:22:22
DateFinished                          : 17/05/2020 08:24:08
TerminatingError                      :
Storage                               : {
  }

WorkflowStatus                        : Completed
Warnings                              : {
  }

ProvisioningSchemeName                : XenPS
MasterImage                           : XDHyp:\HostingUnits\XenHU\Base.vm
    \Base.snapshot
IdentityPoolName                      : idPool1
IdentityPoolUid                       : 03743136-e43b-4a87-af74-
    ab71686b3c16
HostingUnitName                       : XenHU
HostingUnitUid                        : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
CustomProperties                      :
InitialBatchSizeHint                  : 0
ProvisioningSchemeUid                 : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                             : Finished
TaskStateInformation                  :
TaskProgress                          : 100
DiskSize                              : 24
MasterImageNote                       :
```



```
WriteBackCacheDiskSize                : 127
WriteBackCacheMemorySize              : 256
WriteBackCacheDriveLetter             : W
Scopes                                : {
  }

NetworkMaps                           : {
 0 }

ProvisioningSchemeMetadata            : {
  }

ServiceOffering                       :
SecurityGroups                        :
DedicatedTenancy                      : False
ResetAdministratorPasswords           : True
StatusMessageSubstitutions            :
GpuTypeId                             :
UseFullDiskCloneProvisioning          : False
```

EXAMPLE 5

Creates a new provisioning scheme with the name “XenPS2”using the hosting unit “XenHu”and the identity pool “idPool1”from the master VM snapshot called “Base.snapshot”.

The operation runs synchronously, and the return value contains the task details.

```
$provScheme = New-ProvScheme -ProvisioningSchemeName XenPS2 -
    HostingUnitName XenHu -IdentityPoolName idPool1 -CleanOnBoot -
    MasterImageVM XDHyp:\HostingUnits\XenHU\Base.vm\Base.snapshot

$provScheme

TaskId                                : d726222a-04b5-4098-b9ac-
    db85ed9d351b
Type                                  : NewProvisioningScheme
Status                                : Finished
CurrentOperation                      :
TaskExpectedCompletion                :
LastUpdateTime                        : 17/05/2020 08:24:08
ActiveElapsedTime                     : 11
DateStarted                           : 17/05/2020 08:22:22
DateFinished                          : 17/05/2020 08:24:08
TerminatingError                      :
Storage                               : {
  }

WorkflowStatus                        : Completed
Warnings                              : {
  }

```



```
ProvisioningSchemeName              : XenPS2
MasterImage                         : XDHyp:\HostingUnits\XenH\Base.vm\
    Base.snapshot
IdentityPoolName                    : idPool1
IdentityPoolUid                     : 03743136-e43b-4a87-af74-
    ab71686b3c16
HostingUnitName                     : XenHU
HostingUnitUid                      : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
CustomProperties                    :
InitialBatchSizeHint                : 0
ProvisioningSchemeUid               : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                           : Finished
TaskStateInformation                :
TaskProgress                        : 100
DiskSize                            : 24
MasterImageNote                     :
WriteBackCacheDiskSize              : 127
WriteBackCacheMemorySize            : 256
WriteBackCacheDriveLetter           : W
Scopes                              : {
  }

NetworkMaps                         : {
 0 }

ProvisioningSchemeMetadata          : {
  }

ServiceOffering                     :
SecurityGroups                      :
DedicatedTenancy                    : False
ResetAdministratorPasswords         : True
StatusMessageSubstitutions          :
GpuTypeId                           :
UseFullDiskCloneProvisioning        : False
```

EXAMPLE 6

Creates a new provisioning scheme with the name “XenPS2”using the hosting unit “AzureHostingUnit”and the identity pool “idPool1”from the master VM snapshot called “Base.snapshot”and using the machine profile called 1.0.

The operation runs synchronously, and the return value contains the task details.

```
$provScheme = New-ProvScheme -ProvisioningSchemeName XenPS2 -
    HostingUnitName AzureHostingUnit -IdentityPoolName idPool1 -
    CleanOnBoot -MasterImageVM XDHyp:\HostingUnits\AzureHostingUnit\
    image.folder\RG.resourcegroup\masterImage.manageddisk -
    MachineProfile XDHyp:\HostingUnits\AzureHostingUnit\machineprofile.
```



```
    folder\RG.resourcegroup\TS.templatespec\1.0.templatespecversion

$provScheme

TaskId                                : d726222a-04b5-4098-b9ac-
    db85ed9d351b
Type                                  : NewProvisioningScheme
Status                                : Finished
CurrentOperation                      :
TaskExpectedCompletion                :
LastUpdateTime                        : 17/05/2020 08:24:08
ActiveElapsedTime                     : 11
DateStarted                           : 17/05/2020 08:22:22
DateFinished                          : 17/05/2020 08:24:08
TerminatingError                      :
Storage                               : {
  }

WorkflowStatus                        : Completed
Warnings:                             : {
  }

ProvisioningSchemeName                : XenPS2
MachineProfile                        : XDHyp:\HostingUnits\
    AzureHostingUnit\machineprofile.folder\RG.resourcegroup\TS.
    templatespec\1.0.templatespecversion
MasterImage                           : XDHyp:\HostingUnits\
    AzureHostingUnit\image.folder\RG.resourcegroup\masterImage.
    manageddisk
IdentityPoolName                      : idPool1
IdentityPoolUid                       : 03743136-e43b-4a87-af74-
    ab71686b3c16
HostingUnitName                       : AzureHostingUnit
HostingUnitUid                        : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
CustomProperties                      :
InitialBatchSizeHint                  : 0
ProvisioningSchemeUid                 : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                             : Finished
TaskStateInformation                  :
TaskProgress                          : 100
DiskSize                              : 24
MasterImageNote                       :
WriteBackCacheDiskSize                : 127
WriteBackCacheMemorySize              : 256
WriteBackCacheDriveLetter             : W
Scopes                                : {
  }

NetworkMaps                           : {
 0 }

```



```
ProvisioningSchemeMetadata          : {
  }

ServiceOffering                     :
SecurityGroups                      :
DedicatedTenancy                    : False
ResetAdministratorPasswords         : True
StatusMessageSubstitutions          :
GpuTypeId                           :
UseFullDiskCloneProvisioning        : False
```

EXAMPLE 7

Creates a new provisioning scheme with the optional parameter "-ForceCreate"to show how to keep a provisioning scheme that fails in image preparation to assist in troubleshooting the failure.

```
$task = New-ProvScheme -HostingUnitName "xenres" -IdentityPoolName "jf1
    " -InitialBatchSizeHint 1 -MasterImageVM "XDHyp:\HostingUnits\xenres
    \small.vm\Small.snapshot" -NetworkMapping @{
 "0"="XDHyp:\HostingUnits\xenres\my-network.network" }
  -ProvisioningSchemeName "no-vda-test" -ProvisioningSchemeType "MCS" -
     Scope @() -VMCpuCount 1 -VMMemoryMB 256 -ForceCreate
Get-ProvScheme -ProvisioningSchemeName "no-vda-test"


CleanOnBoot                   : False
ControllerAddress             : {
  }

CpuCount                      : 1
DiskSize                      : 1
HostingUnitName               : xenres
HostingUnitUid                : 14d7fa9c-c790-407c-858a-5e5a758a73cc
IdentityPoolName              : jf1
IdentityPoolUid               : 5639673f-1e13-455c-9c41-0f493e24f4e2
MachineCount                  : 0
MachineProfile                :
MasterImageVM                 : XDHyp:\HostingUnits\xenres\small.vm\
    Small.snapshot
MasterImageVMDate             : 9/7/2023 10:14:34 AM
MemoryMB                      : 256
Metadata                      : {
 ImageManagementPrep_DoImagePreparation = True,
     ImageManagementPrep_Excluded_Steps = ,
                                ImageManagementPrep_NoAutoShutdown =
                                    False }

MetadataMap                   : {
 [ImageManagementPrep_DoImagePreparation, True], [
     ImageManagementPrep_Excluded_Steps,
                                ], [ImageManagementPrep_NoAutoShutdown,
```



```
                                    False] }
29
30 PVSSite                        :
31 PVSVDisk                       :
32 PreparedImageDefinitionName    :
33 PreparedImageVersionNumber     :
34 PreparedImageVersionUid        : 00000000-0000-0000-0000-000000000000
35 ProvisioningSchemeName         : no-vda-test
36 ProvisioningSchemeType         : MCS
37 ProvisioningSchemeUid          : 78a38a90-eea3-40bf-a4c8-94c9a1cd6906
38 ProvisioningSchemeVersion      : 1
39 State                          : ErrorCreating
40 TaskId                         :
41 VMMetadata                     : {
42  A, A, E, A... }
43
44 WindowsActivationType          : UnsupportedVDA
45 PersonalVDiskDriveLetter       :
46 PersonalVDiskDriveSize         : 0
47 UsePersonalVDiskStorage        : False
48 NetworkMaps                    : {
49  0 }
50
51 Scopes                         :
52 DedicatedTenancy               : False
53 GpuTypeId                      :
54 ResetAdministratorPasswords    : False
55 SecurityGroups                 : {
56   }
57
58 ServiceOffering                :
59 TenancyType                    : Shared
60 CurrentMasterImageUid          : 20f4e0e1-2983-43c1-b6ba-25e6798f092f
61 CustomProperties               :
62 ImageRuntimeInfo               :
63 UseFullDiskCloneProvisioning   : False
64 UseWriteBackCache              : False
65 WriteBackCacheDiskSize         : 0
66 WriteBackCacheDriveLetter      :
67 WriteBackCacheMemorySize       : 0
68 Warnings                       : {
69   }
70
71 WriteBackCacheDiskIndex        : 0
72
73
74
75 Get-ProvOperationEvent -LinkedObjectUid 78a38a90-eea3-40bf-a4c8-94
      c9a1cd6906
76
77
78 EventAdditionalData : No Image Preparation results found. There may be
      no suitable VDA installed, or some other
```



```
                        serious failure in the Master VM. Image
                           preparation failed.
EventCategory       : Error
EventDateTime       : 9/8/2023 9:53:07 AM
EventId             : 1
EventMessage        :
EventSeverity       : Critical
EventSource         : Mcs
EventState          : New
LinkedObjectType    : ProvisioningScheme
LinkedObjectUid     : 78a38a90-eea3-40bf-a4c8-94c9a1cd6906
OperationName       : Create
OperationTargetName : no-vda-test
OperationTargetType : ProvisioningScheme
OperationType       : ProvisioningSchemeManagement
Recommendation      :
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to be created. This must not be a name that is being used by an existing provisioning scheme, and must not contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitName

The name of the hosting unit to be used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool to be used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitUid

The unique identifier of the hosting unit to be used for the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier of the identity pool to be used for the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionUid

The identifier for the image version used for the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name for the image definition used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionNumber

The version number for the image version used for the provisioning scheme.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MasterImageVM

The path in the hosting unit provider to the VM snapshot or template that will be used. This identifies the hard disk to be used and the default values for the memory and processors. This must be a path to a Snapshot or Template item in the same hosting unit specified by HostingUnitName or HostingUnitUid. Valid paths are of the format:

XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm\<SnapshotName>.snapshot
XDHyp:\HostingUnits\<HostingUnitName>\<path>\<TemplateName>.template

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-VMCpuCount

The number of processors that will be used to create VMs from the provisioning scheme.

Type:	Int32
Position:	Named
Default value:	The number of CPUs assigned to the base image VM snapshot or VM template.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMMemoryMB

The maximum amount of memory that will be used to created VMs from the provisioning scheme in MB.

Type:	Int32
Position:	Named
Default value:	The amount of memory assigned to the base image VM snapshot or VM template.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseWriteBackCache

Indicates whether write-back cache is enabled for the VMs created from this provisioning scheme.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NoImagePreparation

Indicates that image preparation should not be performed on this provisioning scheme.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-NetworkMapping

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, VMs are created with a single NIC, which is mapped to the default network in the hosting unit. If this parameter is supplied, machines are created with the number of NICs specified in the map, and each NIC is attached to the specified network.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

Type:	Hashtable
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FunctionalLevel

The FunctionalLevel of the VDA installed on the given MasterImageVM.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-CleanOnBoot

Indicates whether the VMs created from this provisioning scheme are reset to their initial condition each time they are started.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Scope

The administration scopes to be applied to the new provisioning scheme.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Metadata

The metadata to be associated with this provisioning scheme.

Type:	Hashtable
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceOffering

The Service Offering to use when creating VMs in Cloud Hypervisors.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecurityGroup

The security groups to apply to VMs created in Cloud Hypervisors.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TenancyType

Indicates whether to use tenancy type Shared, Instance or Host when creating VMs in Cloud Hypervisors.

Type:	String
Accepted values:	Host, Instance, Shared
Position:	Named
Default value:	Shared
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineProfile

Currently only supported with Azure. Defines the inventory path to the source VM used by the provisioning scheme as a template. This profile identifies the properties for the VMs created from the scheme. The VM must be in the hosting unit that HostingUnitName or HostingUnitUid refers to. If any properties are present in the MachineProfile but not the CustomProperties, values from the template will be written back to the CustomProperties.

Valid paths are of the format: XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CustomProperties

The properties of the provisioning scheme that are specific to the target hosting infrastructure. See about_ProvCustomProperties for more information.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResetAdministratorPasswords

Indicates whether the passwords for administrator accounts are reset on created machines.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseFullDiskCloneProvisioning

Indicates whether VMs should be created using the dedicated full disk clone feature.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Validate

Indicates whether a dry run of the configuration validation will be performed.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MasterImageNote

A note for the master image.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This can not be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InitialBatchSizeHint

Provides a predictive hint for the number of initial VMs that will be added to the MCS catalog when the scheme is successfully created. Callers should supply this parameter in situations where the completion of New-ProvScheme will be closely followed by a New-ProvVM call to create an initial batch of VMs in the catalog.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PVSSite

PVS Site to be assigned to the VMs in the Provisioning Scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PVSvDisk

PVS vDisk to be assigned to the VMs in the Provisioning Scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeType

The type of Provisioning Scheme to created, either MCS or PVS, defaults to MCS

Type:	ProvisioningSchemeType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ForceCreate

If specified, the provisioning scheme will be created even in the event of creation failure.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For new provisioning scheme tasks, this is always NewProvisioningScheme.

Status <string>

Where in its lifecycle the task is.

CurrentOperation <string>

Operation specific phase of the overall “Running”task state.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateStarted <DateTime>

The date and time when the task was started.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Storage <Citrix.MachineCreation.Sdk.NewProvSchemeStorage[]>

Storage objects to be used for the new provisioning scheme task.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warnings associated with the new provisioning scheme task.

ProvisioningSchemeName <string>

The name of the provisioning scheme being created.

MasterImage <string>

The inventory path of the VM snapshot or template to be used as the master VM image for the task.

IdentityPoolName <string>

The name of the identity pool to be used by the new provisioning scheme.

IdentityPoolUid <guid>

The unique identifier name of the identity pool to be used by the new provisioning scheme.

HostingUnitName <string>

The name of the hosting unit to be used by the new provisioning scheme.

HostingUnitUid <guid>

The unique identifier of the hosting unit to be used by the new provisioning scheme.

CustomProperties <string>

The properties of the provisioning scheme that are specific to the target hosting infrastructure.

InitialBatchSizeHint <int>

The number of VMs that are expected to be added to the scheme as an initial batch.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme being created.

TaskState <Citrix.MachineCreation.Sdk.NewProvisioningSchemeState>

The state of the task. This can be any of the following:

Processing

The task has begun but has not done anything yet.

LocatingResources,

The workflow is locating resources from other services.

HostingUnitNotFound

The task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound

The task failed because the required VM snapshot or VM template could not be located.

ConsolidatingMasterImage

The task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage

The task is replicating the consolidated master image.

StoringProvisioningScheme

The task is storing the provisioning scheme data in the database.

Finished

The task completed with no errors.

ProvisioningSchemeAlreadyExists

The task failed because a provisioning scheme with the same name already exists.

IdentityPoolNotFound

The task failed because the specified identity pool could not be found.

MasterVMImageIsNotPartOfProvisioningSchemeHostingUnit,

The task failed because the hosting unit from which the master image originated is not the same hosting unit that the provisioning scheme is using.

MasterVmImageIsNotASnapshot

The task failed because the master VM path does not refer to a snapshot or VM template item.

ProvisioningSchemeNotFound

The task failed because it could not find a provisioning scheme with the specified name.

TaskAlreadyRunningForProvisioningScheme

The task failed because a task for a provisioning scheme with the same name is already running.

InvalidMasterVMConfiguration

The task failed because the VM snapshot or VM template specified as the master has an invalid configuration.

InvalidMasterVMState

The task failed because the VM snapshot or VM template specified as the master is currently in an invalid state.

InsufficientResources

The task failed because the hypervisor did not have enough resources to complete the task.

DiskConsolidationFailed

The disk consolidation task failed. Details are in the task state information string.

StorageNotFound

The task failed because no associated storage was found in the hosting unit.

ConfigurationError

The task failed because the service is unable to contact one of the other services. This is because not all appropriate Configuration Service registrations have been performed.

RequestedFeatureNotEnabled

The task failed because a requested feature is not enabled.

MachineProfileNotSupported

The task failed because machine profile is not supported.

FailedToReadMachineProfile

Failed to read the Machine Profile.

Canceled

The task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation <string>

Additional information about the current task state.

TaskProgress <double>

The progress of the task 0-100%.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

MasterImageNote <string>

The note of the master image.

WriteBackCacheDiskSize <int>

The size of any write-back cache disk (zero if the write-back cache feature was not selected).

WriteBackCacheMemorySize <int>

The size of the write-back cache (zero if the write-back cache feature was not selected).

WriteBackCacheDriveLetter <char>

The drive letter of write back cache disk (default empty which means to be allocated by operating system).

Scopes <Citrix.Fma.Sdk.ServiceCoreScopeReference[]>

The delegated administration scopes to which the scheme will belong.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The list of NIC to network associations, if specified.

ProvisioningSchemeMetadata <System.Collections.Generic.Dictionary[string, string];>

The metadata to apply to the provisioning scheme, if specified.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

DedicatedTenancy <bool>

Whether to use dedicated tenancy when creating VMs in Cloud Hypervisors.

ResetAdministratorPasswords <bool>

Whether to reset the passwords for administrator accounts on created machines.

StatusMessageSubstitutions <string[]>

List of strings to be substituted into custom error or status messages from provisioning plug-ins.

GpuTypeId <string>

The id of the GPU type used by this provisioning scheme, null if no GPU.

UseFullDiskCloneProvisioning <bool>

Indicates whether the machines are provisioned using the dedicated full disk clone feature.

Notes

The cmdlet is associated with a task of type NewProvisioningScheme, and while active will move through the following operations (CurrentOperation field):

ValidatingInputs

ConsolidatingMasterImage

PreparingMasterImage

ReplicatingMasterImage

CommittingScheme

Only one long-running task for each provisioning scheme can be processed at a time.

In case of failure, the following errors can result.

Error Codes

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

InvalidParameterCombination

Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ScopeNotFound

One or more of the scopes nominated for the new provisioning scheme do not exist.

ImageVersionNotFound

The specified image version for the new provisioning scheme do not exist.

ImageVersionNotReady

The specified image version for the new provisioning scheme is not ready for use.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

VhdParametersMustBeSupplied

When parameter VhdTemplateSource or VhdResultDestination is supplied, both parameters are required to be supplied.

ServiceDoesNotSupportFullDiskClone

The full disk clone parameter is being used when the service does not support the full disk clone feature. Upgrade the service or remove the parameter.

FullDiskCloneDoesNotSupportCleanOnBootVMs

The full disk clone functionality is applicable to dedicated provisioned machines only.

CannotUseWriteBackCacheWithCleanOnBootDisabled

Cannot turn on writebackcache on persistent catalog. Do not use -UseWriteBackCache without specifying -CleanOnBoot.

Related Links

- about_ProvMachineCreationSnapin
- about_Prov_CustomProperties
- Get-ProvTask
- Get-ProvScheme
- Test-ProvSchemeNameAvailable

New-ProvVM

March 11, 2024

Creates new virtual machines.

Syntax

```
New-ProvVM
   -ProvisioningSchemeName <String>
   -ADAccountName <String[]>
   [-FastBuild]
   [-NetworkMapping <Hashtable>]
   [-AutoAssignVLAN]
```



```
    [-SecurityGroup <String[]>]
    [-MachinesPerAssistant <Int32>]
    [-MaxAssistants <Int32>]
    [-RunAsynchronously]
    [-PurgeJobOnSuccess]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
New-ProvVM
   -ProvisioningSchemeName <String>
   -ADAccountSid <String[]>
   [-FastBuild]
   [-NetworkMapping <Hashtable>]
   [-AutoAssignVLAN]
   [-SecurityGroup <String[]>]
   [-MachinesPerAssistant <Int32>]
   [-MaxAssistants <Int32>]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-ProvVM
   -ProvisioningSchemeUid <Guid>
   -ADAccountSid <String[]>
   [-FastBuild]
   [-NetworkMapping <Hashtable>]
   [-AutoAssignVLAN]
   [-SecurityGroup <String[]>]
   [-MachinesPerAssistant <Int32>]
   [-MaxAssistants <Int32>]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
New-ProvVM
   -ProvisioningSchemeUid <Guid>
   -ADAccountName <String[]>
   [-FastBuild]
   [-NetworkMapping <Hashtable>]
   [-AutoAssignVLAN]
   [-SecurityGroup <String[]>]
   [-MachinesPerAssistant <Int32>]
   [-MaxAssistants <Int32>]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Lets you create new virtual machines with the configuration specified by a provisioning scheme.

The virtual machines are created using all of the storage specified in the provisioning scheme. The storage are used in a Round Robin mechanism.

For the duration of this task, the AD accounts are locked by the AD Identity Service, which prevents the same accounts from being used by other machine creation tasks.

Examples

EXAMPLE 1

Creates a new virtual machine using the AD account “domain\account”in the provisioning scheme “MyScheme”.

```
New-ProvVM -ProvisioningSchemeName MyScheme -ADAccountName "domain\
    Account"

TaskId                              : 90e93b9d-a225-4701-ad50-
    fa1546af35ac
Type                                : NewVirtualMachine
Status                              : Finished
CurrentOperation                    :
TaskProgress                        : 100
TaskExpectedCompletion              :
LastUpdateTime                      : 17/05/2020 08:24:08
ActiveElapsedTime                   : 11
DateStarted                         : 17/05/2020 08:22:22
DateFinished                        : 17/05/2020 08:24:08
TerminatingError                    :
WorkflowStatus                      : Completed
MasterImage                         : /Base.vm/Base.snapshot
MasterImageId                       : base-vm\base
ProvisioningSchemeName              : MyScheme
StatusMessageSubstitutions          : {
  }

ProvisioningSchemeUid               : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                           : Finished
TaskStateInformation                :
HostingUnitUid                      : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
HostingUnitName                     : XenHU
```



```
IdentityPoolUid                    : 03743136-e43b-4a87-af74-
    ab71686b3c16
IdentityPoolName                   : idPool1
VirtualMachinesToCreateCount       : 1
VirtualMachinesCreatedCount        : 1
VirtualMachinesCreationFailedCount : 0
CreatedVirtualMachines             : {
 DOMAIN\Account$ }

FailedVirtualMachines              : {
  }
```

EXAMPLE 2

Creates a new virtual machine using the AD Account Sid "S-1-254-569073838-1281390506-2920376713-1253648347-1001"in the provisioning scheme "MyScheme".

```
New-ProvVM -ProvisioningSchemeName MyScheme -ADAccountSid "S
    -1-254-569073838-1281390506-2920376713-1253648347-1001"

TaskId                             : 587717c5-4763-41e1-80c6-1
    d1fb32fa1fd
Type                               : NewVirtualMachine
Status                             : Finished
CurrentOperation                   :
TaskProgress                       : 100
TaskExpectedCompletion             :
LastUpdateTime                     : 17/10/2022 08:24:08
ActiveElapsedTime                  : 11
DateStarted                        : 17/10/2022 08:22:22
DateFinished                       : 17/10/2022 08:24:08
TerminatingError                   :
WorkflowStatus                     : Completed
MasterImage                        : /Base.vm/Base.snapshot
MasterImageId                      : base-vm\base
ProvisioningSchemeName             : MyScheme
StatusMessageSubstitutions         : {
  }

ProvisioningSchemeUid              : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                          : Finished
TaskStateInformation               :
HostingUnitUid                     : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
HostingUnitName                    : XenHU
IdentityPoolUid                    : 03743136-e43b-4a87-af74-
    ab71686b3c16
IdentityPoolName                   : idPool1
VirtualMachinesToCreateCount       : 1
VirtualMachinesCreatedCount        : 1
```



```
VirtualMachinesCreationFailedCount : 0
CreatedVirtualMachines             : {
 Account$ }

FailedVirtualMachines              : {
  }
```

EXAMPLE 3

Creates a new virtual machine using the AD account “domain\account”in the provisioning scheme “MyScheme”asynchronously. To get the task details, use Get-ProvTask -TaskID <task id> .

```
New-ProvVM -ProvisioningSchemeName MyScheme -ADAccountName "domain\
    Account" -RunAsynchronously

Guid
----
6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

Get-ProvTask -TaskID 6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

TaskId                             : 4b49f13a-277c-4cb0-bc40-
    f088430cfe8a
Type                               : NewVirtualMachine
Status                             : Finished
CurrentOperation                   :
TaskProgress                       : 100
TaskExpectedCompletion             :
LastUpdateTime                     : 17/05/2020 08:24:08
ActiveElapsedTime                  : 11
DateStarted                        : 17/05/2020 08:22:22
DateFinished                       : 17/05/2020 08:24:08
TerminatingError                   :
WorkflowStatus                     : Completed
MasterImage                        : /Base.vm/Base.snapshot
MasterImageId                      : base-vm\base
ProvisioningSchemeName             : MyScheme
StatusMessageSubstitutions         : {
  }

ProvisioningSchemeUid              : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                          : Finished
TaskStateInformation               :
HostingUnitUid                     : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
HostingUnitName                    : XenHU
IdentityPoolUid                    : 03743136-e43b-4a87-af74-
    ab71686b3c16
IdentityPoolName                   : idPool1
VirtualMachinesToCreateCount       : 1
```



```
VirtualMachinesCreatedCount         : 1
VirtualMachinesCreationFailedCount : 0
CreatedVirtualMachines              : {
 DOMAIN\Account$ }

FailedVirtualMachines               : {
  }
```

EXAMPLE 4

Creates new virtual machines using all of the available AD accounts from the identity pool “MyPool” in the provisioning scheme “MyScheme”asynchronously. To get the task details, use Get-ProvTask -TaskID <task id>.

```
$accounts = Get-AcctAdAccount -IdentityPool MyPool -State Available
New-ProvVM -ProvisioningSchemeName MyScheme -ADAccountName $t -
    RunAsyncronously
Get-ProvTask -TaskID 6dd85fec-96cf-46b1-9cd4-d8ba7d06e85b

TaskId                              : 4b49f13a-277c-4cb0-bc40-
    f088430cfe8a
Type                                : NewVirtualMachine
Status                              : Finished
CurrentOperation                    :
TaskProgress                        : 100
TaskExpectedCompletion              :
LastUpdateTime                      : 17/05/2020 08:24:08
ActiveElapsedTime                   : 108
DateStarted                         : 17/05/2020 08:22:22
DateFinished                        : 17/05/2020 08:34:08
TerminatingError                    :
WorkflowStatus                      : Completed
MasterImage                         : /Base.vm/Base.snapshot
MasterImageId                       : base-vm\base
ProvisioningSchemeName              : MyScheme
StatusMessageSubstitutions          : {
  }

ProvisioningSchemeUid               : 7585f0de-192e-4847-a6d8-22713
    c3a2f42
TaskState                           : Finished
TaskStateInformation                :
HostingUnitUid                      : 01a4a008-8ce8-4165-ba9c-
    cdf15a6b0501
HostingUnitName                     : XenHU
IdentityPoolUid                     : 03743136-e43b-4a87-af74-
    ab71686b3c16
IdentityPoolName                    : idPool1
VirtualMachinesToCreateCount        : 10
VirtualMachinesCreatedCount         : 10
VirtualMachinesCreationFailedCount : 0
```



```
CreatedVirtualMachines              : {
 DOMAIN\Account1$,... }

FailedVirtualMachines               : {
  }
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme in which the VMs are created.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountName

A list of the Active Directory account names that are used for the VMs. The accounts must be provided in a domain-qualified format. This parameter accepts Identity objects as returned by the New-AcctADAccount cmdlet,

or any PSObject with string properties “Domain”and “ADAccountName”.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ADAccountSid

A list of the Active Directory account sids that are used for the VMs.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme in which the VMs are created.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FastBuild

Indicates whether the command can run faster by using optimizations in the creation process. This may mean that VMs created cannot be started until ResetVM operations have been performed by the Broker and Machine Identity Services.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkMapping

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, provisioned VMs are created with network settings as inherited from the provisioning scheme. If this parameter is supplied, machines are created with the number of NICs specified in the map, and each NIC is attached to the specified network.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AutoAssignVLAN

Indicates whether the VLAN of the network should be automatically assigned to the new VM or if the VLAN from the master image should be used

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SecurityGroup

The security groups to apply to machines created in Cloud Hypervisors.

Type:	String[]
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MachinesPerAssistant

The number of concurrent volume operations that may be performed by each volume worker helper machine.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxAssistants

The maximum number of volume worker help machines that may be created for this operation.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For provisioning new VM tasks, this is always "NewVirtualMachine".

Status <string>

Where in its lifecycle the task is.

CurrentOperation <string>

Operation specific phase of the overall "Running"task state.

TaskProgress <double>

The progress of the task 0-100%.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateStarted <DateTime>

The date and time when the task was started.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

MasterImage <string>

The inventory path of the VM snapshot that was used as the master image for the task.

MasterImageId <string>

The unique identifier of the VM snapshot as assigned by the hosting unit.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the task.

StatusMessageSubstitutions <string[]>

List of strings to be substituted into custom error or status messages from provisioning plug-ins.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the task.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,

Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

HypervisorInMaintenanceMode

Indicates that the hypervisor is not available for normal use.

NoAvailableStorage

Indicates that no storage is available for the hypervisor.

NoAvailableNetwork

Indicates that no network is available for the hypervisor.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

The job failed for reasons specified in TaskStateInformation.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation

Provides more detailed information about the current task state.

HostingUnitUid

The unique identifier of the hosting unit that is used by the provisioning task.

HostingUnitName <string>

The name of the hosting unit that is used by the provisioning task.

IdentityPoolUid <guid>

The unique identifier of the identity pool that is used by the provisioning task.

IdentityPoolName <string>

The name of the identity pool that is used by the provisioning task.

VirtualMachinesToCreateCount <int>

The total number of virtual machines that the task is trying to create.

VirtualMachinesCreatedCount <int>

The number of virtual machines that the task has created so far. Details of the machines that were created are in the CreatedVirtualMachines parameter.

VirtualMachinesCreationFailedCount <int>

The number of virtual machines that the task has failed to create. Details of the machines that were not created are in the FailedVirtualMachines parameter.

CreatedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

Object array of created virtual machines summary information. The object properties are:

ADAccountName <string>

The domain qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents.

Message <string>

The exception message.

Details <string>

The full exception content including stack trace.

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state.

Status <string>

The status of the virtual machine.

StatusAdditionalInformation <string>

Extra information about the Status.

VMId <string>

The virtual machine identifier within the hypervisor in which the VM resides.

VMName <string>

The display name of the virtual machine within the hypervisor in which the VM resides.

FailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

See CreatedVirtualMachines for details of the object parameters.

Notes

The cmdlet is associated with a task of type NewVirtualMachine, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

CreatingVirtualMachines

ReleasingAccountLocks

Only one long-running task in each provisioning scheme can be processed at a time.

In the case of failure, the following errors can result.

Error Codes

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

CommunicationError

An error occurred while communicating with the service.

InvalidParameterCombination

Both PurgeJobOnSuccess and RunAsynchronously were specified.

When running asynchronously, the cmdlet terminates before the job does,

so it cannot clean up the completed job.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Remove-ProvVM

- Lock-ProvVM
- Unlock-ProvVM

Publish-ProvMasterVMImage

March 11, 2024

Update the master image associated with the provisioning scheme.

Syntax

```
Publish-ProvMasterVMImage
        -MasterImageVM <String>
        [-ProvisioningSchemeName] <String>
        [-DoNotStoreOldImage]
        [-VhdTemplateSource <String>]
        [-VhdResultDestination <String>]
        [-AppScanResultsFile <String>]
        [-FunctionalLevel <String>]
        [-MasterImageNote <String>]
        [-RunAsynchronously]
        [-PurgeJobOnSuccess]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Publish-ProvMasterVMImage
        -MasterImageVM <String>
        -ProvisioningSchemeUid <Guid>
        [-DoNotStoreOldImage]
        [-VhdTemplateSource <String>]
        [-VhdResultDestination <String>]
        [-AppScanResultsFile <String>]
        [-FunctionalLevel <String>]
        [-MasterImageNote <String>]
        [-RunAsynchronously]
        [-PurgeJobOnSuccess]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Publish-ProvMasterVMImage
        [-ProvisioningSchemeName] <String>
        [-DoNotStoreOldImage]
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```



```
Publish-ProvMasterVMImage
       -ProvisioningSchemeUid <Guid>
       [-DoNotStoreOldImage]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to update the hard disk image used to create virtual machines. If the provisioning scheme is a 'CleanOnBoot'type, then the next time that virtual machines are started, their hard disks are updated to this new image. Regardless of the 'CleanOnBoot'type, all new virtual machines created after this command will use this new hard disk image.

A background task will be created to remove the old hard disk copies. You can locate and monitor this task using the Get-ProvTask cmdlet.

A snapshot or VM template is used rather than a VM, so that the content of the hard disk for the provisioning scheme can be easily determined.

As the snapshot or VM template are specified using a path into the Citrix Host Service PowerShell Provider, the Citrix Host Service PowerShell snap-in must also be loaded to use this cmdlet.

The previous hard disk image path is stored into the history (see Get-ProvSchemeMasterVMImageHistory). The data stored in the history allows to do a rollback to revert to the previous hard disk image if required.

Examples

EXAMPLE 1

Updates the master hard disk image for the provisioning Scheme "MyScheme"to use the "base.snapshot"hard disk image.

```
Publish-ProvMasterVMImage -ProvisioningSchemeName MyScheme -
    MasterImageVM XDHyp:\HostingUnits\HostUnit1\RhoneCC_baseXP.vm\base.
    snapshot

TaskId                  : 248f102f-073f-45fe-aea9-1819a6d6dd1f
Active                  : False
Host                    : MyHost
DateStarted             : 17/05/2010 17:37:57
Type                    : PublishImage
Metadata                : {
  }

```



```
WorkflowStatus          : Completed
ProvisioningSchemeUid   : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName  : MyScheme
MasterImage             : /RhoneCC_baseXP.vm/base.snapshot
HostingUnitName         : HostUnit1
HostingUnitUid          : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
MasterImageNote         :
ADIdentityPoolName      :
ADIdentityPoolUid       : 03743136-e43b-4a87-af74-ab71686b3c16
CurrentOperation        :
TaskState               : Finished
TaskStateInformation    :
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to which the hard disk image should be updated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-MasterImageVM

The path in the hosting unit provider to the VM snapshot or template that will be used. This identifies the hard disk to be used and the default values for the memory and processors. This must be a path

to a Snapshot or Template item in the same hosting unit used by the provisioning scheme specified by ProvisioningSchemeName or ProvisioningSchemeUid. Valid paths are of the format:

XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm\<SnapshotName>.snapshot
XDHyp:\HostingUnits\<HostingUnitName>\<path>\<TemplateName>.template

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme to which the hard disk image should be updated.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VhdTemplateSource

A file path to a source VHD template to be used when performing application scanning during image preparation. The presence of this parameter in conjunction with VhdResultDestination implies that application scanning is to be performed

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-VhdResultDestination

A file path (including file name) where the VHD disk file containing the results of application scanning should be written.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AppScanResultsFile

File name to which the results of application scanning should be written.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FunctionalLevel

The FunctionalLevel of the VDA installed on the given MasterImageVM.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MasterImageNote

The note of the master image.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the cmdlet should return before it completes. If specified, the command returns an identifier for the task that was created. You can monitor this task using the Get-ProvTask cmdlet.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history will be removed from the database once the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DoNotStoreOldImage

Indicates that the current image should not be added to the image history list for the provisioning scheme. This is useful when rolling back to a previous image.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Publish-ProvMasterVMImage.

Outputs

Guid

When "RunAsynchronously"is specified, this identifier is returned to provide the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Type <Citrix.XDInterServiceTypes.JobType>

The type of the task. For publish master image for provisioning scheme tasks this is always "PublishImage".

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored against the task. For new tasks this will be empty until metadata is added.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is being used to process the task.

ProvisioningSchemeName <string>

The name of the provisioning scheme associated with the task.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme associated with the task.

MasterImage <string>

The path of the VM snapshot or template that was used as the master image for the task.

IdentityPoolName <string>

The name of the identity pool being used for this task.

IdentityPoolUid <guid>

The unique identifier of the identity pool being used for this task.

HostingUnitName <string>

The name of the hosting unit being used for this task.

HostingUnitUid

The unique identifier of the hosting unit being used for this task.

MasterImageNote

The note of the master image.

TaskState <Citrix.DesktopUpdateManager.SDK.NewProvisioningSchemeState>

The state of the task. This can be any of the following:

Processing

Indicates that the task has just begun and has not done anything yet.

LocatingResources,

Indicates that the workflow is locating resources from other services.

HostingUnitNotFound

Indicates that the task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound

Indicates that the task failed because the required snapshot or VM template could not be located.

ConsolidatingMasterImage

Indicates that the task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage

Indicates that the task is replicating the consolidated master image.

StoringProvisioningScheme

Indicates that the task is storing the provisioning scheme data to the database.

Finished

Indicates that the task has completed with no errors.

ProvisioningSchemeAlreadyExists

Indicates that the task failed because a provisioning scheme with the same name already exists.

IdentityPoolNotFound

Indicates that the task failed because the identity pool specified could not be found.

MasterVMImageIsNotPartOfProvisioningSchemeHostingUnit,

Indicates that the hosting unit that the master image originated from is not the same hosting unit that the provisioning scheme is defined to use.

MasterVmImageIsNotASnapshot

Indicates that the task failed because the master VM image path does not refer to a snapshot or a VM template item.

ProvisioningSchemeNotFound

Could not find a provisioning scheme with the specified name.

TaskAlreadyRunningForProvisioningScheme

A task for a provisioning scheme with the same name is already running.

InvalidMasterVMConfiguration

The VM snapshot or VM template specified as the master had an invalid configuration.

InvalidMasterVMState

The VM snapshot or VM template specified as the master is currently in an invalid state.

InsufficientResources

Indicates that the task failed because the hypervisor did not have enough resources to complete the task.

StorageNotFound

Indicates that no associated storage was found in the hosting unit.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation <string>

Holds string data providing more details about the current task state.

TaskProgress <double>

The progress of the task 0-100%.

Notes

The cmdlet is associated with a task of type PublishImage, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

ConsolidatingMasterImage

PreparingMasterImage

ReplicatingMasterImage

CommittingScheme

Only one long-running task per provisioning scheme can be processed at any one time.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeMasterVMImageHistory
- Get-ProvScheme

Remove-ProvImageDefinition

March 11, 2024

Remove an image definition.

Syntax

```
Remove-ProvImageDefinition
      -PreparedImageDefinitionName <String>
      [-PurgeDBOnly]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageDefinition
      -PreparedImageDefinitionUid <Guid>
      [-PurgeDBOnly]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove an image definition and its image versions. The image definition can contain multiple image versions. If it still has image versions used by provisioning scheme, the image scheme can not be removed.

Examples

EXAMPLE 1

Remove the image definition by name and its image versions.

```
Remove-ProvImageDefinition -PreparedImageDefinitionName MyImage
```

Parameters

-PreparedImageDefinitionName

The name of the image definition to be removed.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreparedImageDefinitionUid

The unique identifier for the image definition to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PurgeDBOnly

If this option is specified, this command will only remove the image definition data from the Citrix site database. However, the disk images created by its image versions still remain in the hypervisor. The hypervisor administrator can remove hard disk images using the tools provided by the hypervisor itself. This option can also be used when you are no longer able to contact the hypervisor and you want to remove the image definition from Citrix site database. This parameter cannot be used with "ForgetVM".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether or not the cmdlet should return before it is complete. If specified, the command returns an identifier for the task that was created. You can monitor this task using the Get-ProvTask cmdlet.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history will be removed from the database when the task has finished. This cannot be specified for tasks that are run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageDefinition

You can pipe an object containing a parameter called 'PreparedImageDefinitionName'or 'Prepared-ImageDefinitionUid'to Remove-ProvImageDefinition.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information: TaskId <Guid> The identifier for the task that was performed. Active <bool> Indicates whether the task is still processing or is complete. Host <string> The name of the host on which the task is running or was run. DateStarted <DateTime> The date and time when the task was started. Metadata

<Citrix.MachineCreation.Sdk.Metadata[]> The list of metadata stored for the task. For new tasks, this is empty until metadata is added. ProvImageDefinitionUid <Guid> The unique identifier of the image definition. ProvImageDefinitionName <string> The name of the image definition. CurrentOperation <string> Operation specific phase of the overall "Running"task state. TaskProgress <double> The progress of the task 0-100%. LastUpdateTime <DateTime> The date and time of the last task status update. ActiveElapsedTime <int> Number of seconds the task has taken for active execution. DateFinished <DateTime> The date and time when the task was completed. TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError> Diagnostic information if the task completely fails. Type <Citrix.XDInterServiceTypes.JobType> The type of task. For new reset disk tasks, this is always "Reset-ProvVMDisk". Status <string> TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState> The state of the task. This can be any of the following:

Processing Indicates that the task is in its initial state. LocatingResources, Indicates that the task is locating information from other services. HostingUnitNotFound Indicates that the required hosting unit could not be located. ImageDefinitionNotFound Indicates that the required image definition could not be located. Finished Indicates that the task is complete. FinishedWithErrors Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine. RemovingImageVersion Indicates that the task is removing image version. RemovingImageDefinition Indicates that the task is removing image definition. Failed Job failed for reasons specified in TaskStateInformation. Canceled Indicates that the task was stopped by user intervention (using Stop-ProvTask) TaskStateInformation <string> Provides more detailed information about the current task state. WorkflowStatus <System.Workflow.Runtime.WorkflowStatus> Indicates the status of the workflow that is used to process the task. TaskExpectedCompletion <DateTime> The date and time at which the task is expected to complete.

Notes

In the case of failure, the following errors can result.

Error Codes

IllegalParameter

One or more parameters are illegal or are not specified.

ImageDefinitionNotFound

The specified image definition could not be located.

UnableToRemoveImageDefinitionDueToAssociatedProvisioningScheme

The image definition still has image versions used by provisioning scheme and can not be removed.

UnableToRemoveImageDefinitionWithPreparingImageVersion

The image definition has image version in preparing and can not be removed.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageDefinition
- New-ProvImageDefinition
- Rename-ProvImageDefinition
- Set-ProvImageDefinition

Remove-ProvImageScheme

March 11, 2024

Removes an image scheme

Syntax

```
Remove-ProvImageScheme
      -PreparedImageSchemeName <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageScheme
      -PreparedImageSchemeUid <Guid>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove an image Scheme. The image scheme may associate with image versions. If associated image versions exist, the image scheme can not be removed.

Examples

EXAMPLE 1

Remove the image scheme by name.

```
Remove-ProvImageScheme -PreparedImageSchemeName azurescheme
```

Parameters

-PreparedImageSchemeName

The name of the image scheme to be removed.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-PreparedImageSchemeUid

The unique identifier for the image scheme to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageScheme

You can pipe an object containing a parameter called 'ImageSchemeName'or 'ImageSchemeUid'to Remove-ImageScheme.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

IllegalParameter

One or more parameters are illegal or are not specified.

ImageSchemeNotFound

The specified image scheme could not be located.

UnableToRemoveImageSchemeDueToAssociatedImageVersion

The image scheme associated with image versions and can not be removed.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageScheme
- New-ProvImageScheme

Remove-ProvImagesPendingDelete

March 11, 2024

Remove pending image to delete.

Syntax

```
Remove-ProvImagesPendingDelete
      [-Id] <Int32>
      [-ForceDelete]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to remove image which are pending to be delete

Examples

EXAMPLE 1

Trying to delete a image which is in use

```
Remove-ProvImagesPendingDelete -Id 1 -ForceDelete
Remove-ProvImagesPendingDelete : Image is in use.
At line:1 char:1
+ Remove-ProvImagesPendingDelete -PendingDeleteId 1 -ForceDelete
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
+ CategoryInfo          : InvalidOperation: (:) [Remove-
    ProvImagesPendingDelete], InvalidOperationException
+ FullyQualifiedErrorId : Citrix.XDPowerShell.MachineCreationStatus.
    DiskIsInUse,Citrix.MachineCreation.Sdk.Commands.
    RemoveProvImagesPendingDeleteCommand
```

EXAMPLE 2

Trying to delete the image which does not exits

```
Remove-ProvImagesPendingDelete -Id 3 -ForceDelete
Remove-ProvImagesPendingDelete : There is no pending image to delete.
At line:1 char:1
+ Remove-ProvImagesPendingDelete -PendingDeleteId 3 -ForceDelete
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
+ CategoryInfo          : InvalidOperation: (:) [Remove-
    ProvImagesPendingDelete], InvalidOperationException
+ FullyQualifiedErrorId : Citrix.XDPowerShell.MachineCreationStatus.
    NoPendingDiskToDelete,Citrix.MachineCreation.Sdk.Commands.
    RemoveProvImagesPendingDeleteComman
d
```

Parameters

-Id

Pendig image record ID, in DB

Type: Int32

Position:	2
Default value:	0
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ForceDelete

If this switch is enabled, we force delete the image from hypervisor

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Related Links

- Get-ProvImagesPendingDelete

Remove-ProvImageVersion

March 11, 2024

Remove specified image version.

Syntax

```
Remove-ProvImageVersion
      -PreparedImageDefinitionName <String>
      -PreparedImageVersionNumber <String>
      [-PurgeDBOnly]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageVersion
      -PreparedImageVersionUid <Guid>
      [-PurgeDBOnly]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove an image version and its relevant hard disk image that has been prepared to provision virtual machines.

A background task will be created to remove the old hard disk copies. You can locate and monitor this task using the Get-ProvTask cmdlet.

Examples

EXAMPLE 1

Removes image version with Guid 90e93b9d-a225-4701-ad50-fa1546af35ac and its prepared master hard disk image.

```
Remove-ProvImageVersion -PreparedImageVersionUid 90e93b9d-a225-4701-
    ad50-fa1546af35ac
```

EXAMPLE 2

Removes image version v1.0 in image1 and its prepared master hard disk image.

```
Remove-ProvImageVersion -PreparedImageVersionNumber 1 -
    PreparedImageDefinitionName image1
```


Parameters

-PreparedImageDefinitionName

The name of image definition which the image version belongs to.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreparedImageVersionNumber

The version number of image version.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreparedImageVersionUid

The identifier of image version.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-PurgeDBOnly

If this option is specified, this command will only remove the image version data from the Citrix site database. However, the disk images created in the image version still remain in the hypervisor. The hypervisor administrator can remove hard disk images using the tools provided by the hypervisor itself. This option can also be used when you are no longer able to contact the hypervisor and you want to remove the image version from Citrix site database. This parameter cannot be used with "ForgetVM" .

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If this is specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageVersion

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

IllegalParameter

One or more parameters are illegal or are not specified.

ImageVersionNotFound

The specified image version could not be located.

UnableToRemoveImageVersionInPreparing

Unable to remove an image version in preparing

UnableToRemoveImageVersionDueToAssociatedProvisioningScheme

Unable to remove specified image version due to associated provisioning scheme.

UnableToRemoveImageVersionDueToAssociatedVirtualMachine

Unable to remove specified image version due to associated virtual machine.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageVersion
- New-ProvImageVersion
- Set-ProvImageVersion

Remove-ProvImageVersionMetadata

March 11, 2024

Removes metadata from the given image version.

Syntax

```
Remove-ProvImageVersionMetadata
      [-PreparedImageVersionUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ProvImageVersionMetadata
      [-PreparedImageVersionUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageVersionMetadata
      [-PreparedImageDefinitionName] <String>
      [-PreparedImageVersionNumber] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageVersionMetadata
      [-PreparedImageDefinitionName] <String>
      [-PreparedImageVersionNumber] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageVersionMetadata
      [-InputObject] <ImageVersion[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvImageVersionMetadata
      [-InputObject] <ImageVersion[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given image version.

Examples

EXAMPLE 1

Remove all metadata from all image version objects.


```
Get-ProvImageVersion | % {
  Remove-ProvImageVersionMetadata -Map $_.MetadataMap  }
```

Parameters

-PreparedImageVersionUid

Id of the image version

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-PreparedImageDefinitionName

Name of the image definition

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-PreparedImageVersionNumber

Version number of the image version

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	ImageVersion[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “Sys-

tem.Collections.Generic.Dictionary[string, string];"). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageVersion

You can pipe a ImageVersion object or any object containing a parameter called 'PreparedImageDefinitionName'and 'PreparedImageVersionNumber'or 'PreparedImageVersionUid'to Remove-ProvImageVersionMetadata.

PSObject

A metadata map object can be piped to the Remove-ProvImageVersionMetadata command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes ————InvalidParameterCombination

The cmdlet parameters are inconsistent.

ObjectNotFound

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvImageVersionMetadata
- Set-ProvImageVersionMetadata

Remove-ProvMetadataConfiguration

March 11, 2024

Remove VM metadata configuration settings for a plugin.

Syntax

```
Remove-ProvMetadataConfiguration
      [-PluginType] <String>
      [-ConfigurationName] <String>
      [-ConfigurationValue] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove configure metadata setting.

Examples

EXAMPLE 1

Remove Azure VM extension AADLoginForWindows from the supported list. Only customer added item can be removed.

```
Remove-ProvMetadataConfiguration -PluginType "AzureRmFactory" -
    ConfigurationName "Extension" -ConfigurationValue "
    AADLoginForWindows"
```

Parameters

-PluginType

The name of the hypervisor plug-in factory. Currently, AzureRmFactory is the only supported plug-in factory.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ConfigurationName

The configuration name. Currently, Extension is the only supported configuration.

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ConfigurationValue

The value for metadata configuration, for example Azure extension type.

Type:	String
Position:	4
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ConfigurationDoesNotExist

The configuration cannot be found in the database.

ConfigurationCitrixDefined

The user cannot remove Citrix defined configurations.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details, check the Windows event logs on your self-hosted delivery controller or contact Citrix support if using Citrix DaaS (Citrix-hosted delivery controller).

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvMetadataConfiguration
- Add-ProvMetadataConfiguration

Remove-ProvOperationEvent

March 11, 2024

Removes one or more MCS operation events.

Syntax

```
Remove-ProvOperationEvent
      [-EventId <Int32[]>]
      [-LinkedObjectType <String>]
      [-LinkedObjectUid <Guid>]
      [-All]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Allows you to remove a set of existing MCS operation events.

Examples

EXAMPLE 1

Removes all of the operation events of the provisioning scheme with id 7585f0de-192e-4847-a6d8-22713c3a2f42.

```
Remove-ProvOperationEvent -LinkedObjectType ProvisioningScheme -
    LinkedObjectUid 7585f0de-192e-4847-a6d8-22713c3a2f42
```

Parameters

-EventId

The array of event id which all associated operation events will be removed.

Type:	Int32[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LinkedObjectType

The type of the linked object for which all associated operation events will be removed.

Type:	String
Accepted values:	ProvisioningScheme
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LinkedObjectUid

The unique identifier of the linked object for which all associated operation events will be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-All

All operation events will be removed.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvOperationEvent

Remove-ProvScheme

March 11, 2024

Removes a provisioning scheme

Syntax

```
Remove-ProvScheme
      [-ProvisioningSchemeName] <String>
      [-PurgeDBOnly]
      [-ForgetVM]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvScheme
      -ProvisioningSchemeUid <Guid>
      [-PurgeDBOnly]
      [-ForgetVM]
      [-RunAsynchronously]
      [-PurgeJobOnSuccess]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to remove a provisioning Scheme. The provisioning scheme must not contain any VMs unless the 'ForgetVM'or 'PurgeDBOnly'option is specified. Use the Get-ProvTask command to monitor the progress of this task.

Examples

EXAMPLE 1

Remove the empty provisioning scheme by name.

```
Remove-ProvScheme -ProvisioningSchemeName $provScheme.
    ProvisioningSchemeName
```

EXAMPLE 2

Deletes provisioning scheme 'Catalog-1'data from the Citrix site database

```
Get-ProvScheme -ProvisioningSchemeName Catalog-1 | Remove-ProvScheme -
    PurgeDBOnly

TaskId                 : 02693f9c-8fa0-41c8-ab23-15853537fbb1
Active                 : False
Host                   : E73755-92-1
DateStarted            : 8/19/2022 12:50:34 PM
Metadata               :
ProvisioningSchemeName : Catalog-1
ProvisioningSchemeUid  : 60851748-3476-43c7-b406-c25f1a952185
PurgeDBOnly            : True
ForgetVm               : False
RemovedVirtualMachines : {
  }

FailedVirtualMachines  : {
  }

TaskState              : Finished
TaskStateInformation   :
CurrentOperation       :
TaskProgress           : 100
LastUpdateTime         : 8/19/2022 12:50:36 PM
ActiveElapsedTime      : 0
DateFinished           : 8/19/2022 12:50:36 PM
TerminatingError       :
Type                   : RemoveProvScheme
Status                 : Finished
WorkflowStatus         : Completed
```



```
TaskExpectedCompletion :
```

EXAMPLE 3

Removes all the Citrix-assigned identifiers(like tags or custom-attributes) on provisioning scheme, VMs and their related resources from hypervisor and also the deletes provisioning scheme 'Catalog-1' data from the Citrix site database.

```
Get-ProvScheme -ProvisioningSchemeName Catalog-1 | Remove-ProvScheme -
    ForgetVM

TaskId                 : 02693f9c-8fa0-41c8-ab23-15853537fbb1
Active                 : False
Host                   : E73755-92-1
DateStarted            : 8/19/2022 12:50:34 PM
Metadata               :
ProvisioningSchemeName : Catalog-1
ProvisioningSchemeUid  : 60851748-3476-43c7-b406-c25f1a952185
PurgeDBOnly            : False
ForgetVm               : True
RemovedVirtualMachines : {
 S-1-5-21-12345678-1234567890-123456789-2345 }

FailedVirtualMachines  : {
  }

TaskState              : Finished
TaskStateInformation   :
CurrentOperation       :
TaskProgress           : 100
LastUpdateTime         : 8/19/2022 12:50:36 PM
ActiveElapsedTime      : 0
DateFinished           : 8/19/2022 12:50:36 PM
TerminatingError       :
Type                   : RemoveProvScheme
Status                 : Finished
WorkflowStatus         : Completed
TaskExpectedCompletion :
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to be removed.

Type: String

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PurgeDBOnly

If this option is specified, this command will only remove the provisioning scheme data from the Citrix site database. However, the VMs and related resources created in the provisioning scheme still remain in the hypervisor. The hypervisor administrator can remove the VMs and hard disk images using the tools provided by the hypervisor itself. This option can also be used when you are no longer able to contact the hypervisor and you want to remove the provisioning scheme from Citrix site database. This parameter cannot be used with "ForgetVM".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ForgetVM

If this option is specified, this command will remove the provisioning scheme data from the Citrix site database and also delete Citrix-assigned identifiers(like tags or custom-attributes) on provisioning scheme, VMs and their related resources from hypervisor. However, the VMs and related resources created in the provisioning scheme still remain in the hypervisor after removing identifiers(like tags or custom-attributes). The hypervisor administrator can remove the VMs and hard disk images using the tools provided by the hypervisor itself. This parameter is only applied to persistent VMs and cannot be used with "PurgeDBOnly".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If this is specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Remove-ProvScheme.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information: TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored for the task. For new tasks, this is empty until metadata is added.

CurrentOperation <string>

Operation specific phase of the overall "Running"task state.

TaskProgress <double>

The progress of the task 0-100%.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For new reset disk tasks, this is always "Reset-ProvVMDisk".

Status <string> ProvisioningSchemeName <string>

The name of the provisioning scheme from which the VM was created.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme from which the VM was created.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,

Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

Job failed for reasons specified in TaskStateInformation.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>

Provides more detailed information about the current task state.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

Notes

If the hosting unit referenced by the provisioning scheme no longer exists (that is, has been removed using the Hosting Unit PowerShell snap-in), the provisioning scheme data is deleted from the database without errors. However, the hard disks associated with the provisioning scheme cannot be removed and remain in the hypervisor.

In the case of failure, the following errors can result.

Error Codes

IllegalParameter

One or more parameters are illegal or are not specified.

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

UnableToRemoveProvisioningSchemeDueToAssociatedVM

The provisioning scheme contained VMs and the 'ForgetVM' or 'PurgeDBOnly' parameter was not specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

The cmdlet is associated with a task of type DisusedImageCleanUp, and while active will move through the following operations (CurrentOperation field)

Running

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvTask
- New-ProvScheme

Remove-ProvSchemeControllerAddress

March 11, 2024

Removes controller addresses from a provisioning scheme.

Syntax

```
Remove-ProvSchemeControllerAddress
      [-ProvisioningSchemeName] <String>
      [-ControllerAddress] <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeControllerAddress
      -ProvisioningSchemeUid <Guid>
      [-ControllerAddress] <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Removes the specified controller addresses from the specified provisioning scheme.

Examples

EXAMPLE 1

Remove all controller addresses from the provisioning scheme with the name “scheme1”.

```
Get-ProvScheme -ProvisioningSchemeName scheme1 | Remove-
   ProvSchemeControllerAddress

CleanOnBoot                 : True
ControllerAddress           : {
  }

CpuCount                    : 1
DiskSize                    : 20
HostingUnitName             : HostUnit1
HostingUnitUid              : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName            : idPool1
IdentityPoolUid             : 03743136-e43b-4a87-af74-ab71686b3c16
```



```
MachineCount                  : 0
MachineProfile                :
MasterImageVM                 : Base.vm/base.snapshot
MasterImageVMDate             : 17/05/2020 09:53:40
MemoryMB                      : 1024
Metadata                      : {
 Department = Sales }

MetadataMap                   : {
 [Department = Sales] }

ProvisioningSchemeUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeName        : Scheme2
ProvisioningSchemeVersion     : 1
TaskId                        : 00000000-0000-0000-0000-000000000000
VMMetadata                    : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter      :
PersonalVDiskDriveSize        : 0
UsePersonalVDiskStorage       : False
NetworkMaps                   : {
 0 }

Scopes                        :
DedicatedTenancy              : False
GpuTypeId                     :
ResetAdministratorPasswords   : False
SecurityGroups                : {
  }

ServiceOffering               :
TenancyType                   : Shared
AzureAdJoinType               :
CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
CustomProperties              :
IdentityType:                 : ActiveDirectory
UseFullDiskCloneProvisioning  : False
UseWriteBackCache             : True
WriteBackCacheDiskSize        : 24
WriteBackCacheMemorySize      : 256
Warnings                      : {
  }

WriteBackCacheDiskIndex       : 0
```

EXAMPLE 2

Remove a subset of the controller address list from the provisioning scheme with the identifier "01a4a008-8ce8-4165-ba9c-cdf15a6b0501".


```
Remove-ProvSchemeControllerAddress -ProvisioningSchemeUid "01a4a008-8ce8-4165-ba9c-cdf15a6b0501" -ControllerAddress (ddcA.citrix.com,ddcC.citrix2.com)

CleanOnBoot                  : True
ControllerAddress            : {
 ddcB.citrix.com }

CpuCount                     : 1
DiskSize                     : 20
HostingUnitName              : HostUnit1
HostingUnitUid               : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName             : idPool1
IdentityPoolUid              : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                 : 0
MachineProfile               :
MasterImageVM                : Base.vm/base.snapshot
MasterImageVMDate            : 17/05/2020 09:53:40
MemoryMB                     : 1024
Metadata                     : {
 Department = Sales }

MetadataMap                  : {
 [Department = Sales] }

ProvisioningSchemeName       : Scheme2
ProvisioningSchemeUid        : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
ProvisioningSchemeVersion    : 1
TaskId                       : 00000000-0000-0000-0000-000000000000
VMMetadata                   : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter     :
PersonalVDiskDriveSize       : 0
UsePersonalVDiskStorage      : False
NetworkMaps                  : {
 0 }

Scopes                       :
DedicatedTenancy             : False
GpuTypeId                    :
ResetAdministratorPasswords  : False
SecurityGroups               : {
  }

ServiceOffering              :
TenancyType                  : Shared
AzureAdJoinType              :
CurrentMasterImageUid        : c0571690-4f57-4476-901b-fe64d6aecb79
CustomProperties             :
IdentityType:                : ActiveDirectory
UseFullDiskCloneProvisioning : False
UseWriteBackCache            : True
```



```
WriteBackCacheDiskSize        : 24
WriteBackCacheMemorySize      : 256
Warnings                      : {
  }

WriteBackCacheDiskIndex       : 0
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme that controller addresses are to be removed from.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme that controller addresses are to be removed from.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerAddress

Specifies the array of controller DNS names to be removed from the provisioning scheme.

Type:	String[]
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

Type:	String[]
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Remove-ProvSchemeControllerAddress.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ControllerAddressNotFound

The specified address was not associated with the provisioning scheme.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvScheme
- Add-ProvSchemeControllerAddress

Remove-ProvSchemeMasterVMImageHistory

March 11, 2024

Removes the history of provisioning scheme master image VMs.

Syntax

```
Remove-ProvSchemeMasterVMImageHistory
      [-ProvisioningSchemeName] <String>
      [-VMImageHistoryUid <Guid>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ProvSchemeMasterVMImageHistory
      -ProvisioningSchemeUid <Guid>
      [-VMImageHistoryUid <Guid>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeMasterVMImageHistory
      -VMImageHistoryUid <Guid>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove the record of previously used master image VMs or snapshots for provisioning schemes.

Examples

EXAMPLE 1

Removes all history for the provisioning scheme called "MyScheme".

```
Remove-ProvSchemeMasterVMImageHistory -ProvisioningSchemeName MyScheme
```

EXAMPLE 2

Removes all history for all provisioning schemes.

```
Get-ProvScheme | Remove-ProvSchemeMasterVMImageHistory
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-VMImageHistoryUid

The unique identifier of the image history item.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

Type:	Guid
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Remove-ProvSchemeMasterVMImageHistory.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeMasterVMImageHistory
- Publish-ProvMasterVMImage

Remove-ProvSchemeMetadata

March 11, 2024

Removes metadata from the given provisioning scheme.

Syntax

```
Remove-ProvSchemeMetadata
      [-ProvisioningSchemeUid] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeMetadata
      [-ProvisioningSchemeUid] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeMetadata
      [-ProvisioningSchemeName] <String>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeMetadata
      [-ProvisioningSchemeName] <String>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeMetadata
      [-InputObject] <ProvisioningScheme[]>
      -Name <String>
```



```
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Remove-ProvSchemeMetadata
        [-InputObject] <ProvisioningScheme[]>
        -Map <PSObject>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given provisioning scheme.

Examples

EXAMPLE 1

Remove all metadata from all provisioning scheme objects.

```
Get-ProvScheme | % {
  Remove-ProvSchemeMetadata -Map $_.MetadataMap  }
```

Parameters

-ProvisioningSchemeUid

Id of the provisioning scheme

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeName

Name of the provisioning scheme

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

Objects to which the metadata is to be added.

Type:	ProvisioningScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe a ProvisioningScheme object or any object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Remove-ProvSchemeMetadata.

PSObject

A metadata map object can be piped to the Remove-ProvSchemeMetadata command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvSchemeMetadata
- Set-ProvSchemeMetadata

Remove-ProvSchemeScope

March 11, 2024

Remove the specified provisioning scheme(s) from the given scope(s).

Syntax


```
Remove-ProvSchemeScope
      [-Scope] <String[]>
      -InputObject <ProvisioningScheme[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeScope
      [-Scope] <String[]>
      -ProvisioningSchemeUid <Guid[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeScope
      [-Scope] <String[]>
      -ProvisioningSchemeName <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

The Remove-ProvSchemeScope command is used to remove one or more provisioning schemes from the given scope(s).

To remove a provisioning scheme from a scope you need permission to change the scopes of the provisioning scheme.

If the provisioning scheme is not in a specified scope, that scope will be silently ignored.

Examples

EXAMPLE 1

Removes a single provisioning scheme from the 'Finance'scope.

```
Remove-ProvSchemeScope Finance -ProvisioningSchemeUid 6702C5D0-C073
    -4080-A0EE-EC74CB537C52
```

EXAMPLE 2

Removes a single provisioning scheme from multiple scopes.

```
Remove-ProvSchemeScope Finance,Marketing -ProvisioningSchemeUid 6702
    C5D0-C073-4080-A0EE-EC74CB537C52
```


EXAMPLE 3

Removes all visible provisioning schemes from the 'Finance'scope.

```
Get-ProvScheme | Remove-ProvSchemeScope Finance
```

EXAMPLE 4

Removes provisioning schemes whose name starts with an 'A'from the 'Finance'scope.

```
Remove-ProvSchemeScope Finance -ProvisioningSchemeName A*
```

Parameters

-Scope

Specifies the scopes to remove the objects from.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

Specifies the ProvisioningScheme objects to be removed.

Type:	ProvisioningScheme[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

Specifies the ProvisioningScheme objects to be removed by the unique identifier of the provisioning scheme(s).

Type:	Guid[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeName

Specifies the ProvisioningScheme objects to be removed by the name of the provisioning scheme(s).

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe a ProvisioningScheme object or any object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Remove-ProvScope.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

ScopeNotFound

One of the specified scopes was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command with the specified objects or scopes.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvSchemeScope
- Get-ProvScopedObject

Remove-ProvSchemeVersion

March 11, 2024

Removes a provisioning scheme configuration version

Syntax

```
Remove-ProvSchemeVersion
      [-ProvisioningSchemeName] <String>
      -Version <Int32>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeVersion
      -ProvisioningSchemeUid <Guid>
      -Version <Int32>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove a prior provisioning scheme configuration version.

Examples

EXAMPLE 1

Removes version 3 of provisioning scheme MyScheme.

```
Remove-ProvSchemeVersion -ProvisioningSchemeName MyScheme -Version 3
```

EXAMPLE 2

Removes version 1 of provisioning scheme with Uid 5b37b311-fa3f-49dd-b93b-661b9e6fa571.

```
Remove-ProvSchemeVersion -ProvisioningSchemeUid 5b37b311-fa3f-49dd-b93b
   -661b9e6fa571 -Version 1
```


Parameters

-ProvisioningSchemeName

The name of the provisioning scheme with the version to be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme with the version to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-Version

The version to remove. Cannot be the current provisioning scheme version in use or a version referenced by any machine in the provisioning scheme.

Type:	Int32
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Remove-ProvSchemeVersion.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound The specified provisioning scheme could not be located.

ProvisioningSchemeVersionNotFound The specified ProvisioningScheme version could not be located.

ProvisioningSchemeVersionRemovalConflict The provisioning scheme version cannot be removed due to it being referenced by other resources.

CannotRemoveCurrentProvisioningSchemeVersion The provisioning scheme version cannot be removed since it is the current version.

DatabaseError An error occurred in the service while attempting a database operation.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

CommunicationError An error occurred while communicating with the service.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeVersion
- Set-ProvScheme

Remove-ProvSchemeWarning

March 11, 2024

Removes one or more provisioning scheme warnings.

Syntax

```
Remove-ProvSchemeWarning
      [-ProvisioningSchemeName <String>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeWarning
      [-ProvisioningSchemeUid <Guid>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeWarning
      [-WarningId <Int32>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvSchemeWarning
      [-All]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Allows you to remove a set of existing provisioning scheme warnings.

Examples

EXAMPLE 1

Removes all of the provisioning scheme warnings matching the specified ProvisioningSchemeUid.

```
Remove-ProvSchemeWarning -ProvisioningSchemeUid 7585f0de-192e-4847-a6d8
    -22713c3a2f42
```


Parameters

-ProvisioningSchemeName

The name of the provisioning scheme for which all associated warnings will be removed.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme for which all associated warnings will be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningId

The id of the warning to remove.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-All

Remove all warnings.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeWarning
- Set-ProvSchemeWarning

Remove-ProvServiceConfigurationData

March 11, 2024

Removes configuration data from the service.

Syntax

```
Remove-ProvServiceConfigurationData
      [[-Name] <String>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove data item from the Machine Creation Service configuration data.

Examples

EXAMPLE 1

Removes the configuration data item with name “MyName”.

```
Remove-ProvServiceConfigurationData -Name "MyName"
```

Parameters

-Name

The name of the configuration data item to remove.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvServiceConfigurationData
- Get-ProvServiceConfigurationData

Remove-ProvServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-ProvServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
```



```
5         [<CitrixCommonParameters>]
6         [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
1  Get-ProvService | % {
2    Remove-ProvServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvServiceMetadata

Remove-ProvTask

March 11, 2024

Removes completed tasks from the database for the Machine Creation Service.

Syntax

```
Remove-ProvTask
       [-TaskId] <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Enables completed tasks that have run within the Machine Creation Service to be removed from the database.

Examples

EXAMPLE 1

Remove entries for all completed tasks from the Machine Creation Service.

```
Get-ProvTask -active $false | Remove-ProvTask
Success
```

Parameters

-TaskId

Specifies the identifier for the task to be removed.

Type:	Guid
Position:	2
Default value:	None
Required:	True

Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Remove-ProvTask command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can result.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

InvalidWorkflow

The specified task could not be found.

InvalidWorkflowState

The task was not in an appropriate state for the requested operation.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvTask
- Add-ProvTaskMetadata
- Remove-ProvTaskMetadata

Remove-ProvTaskMetadata

March 11, 2024

Removes metadata from the given Task.

Syntax

```
Remove-ProvTaskMetadata
      [-TaskId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvTaskMetadata
      [-TaskId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-ProvTaskMetadata
      [-InputObject] <Task[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-ProvTaskMetadata
      [-InputObject] <Task[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Task.

Examples

EXAMPLE 1

Remove all metadata from all Task objects.

```
Get-ProvTask | Remove-ProvTaskMetadata
```

Parameters

-TaskId

Id of the Task.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	Task[]

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Remove-ProvTaskMetadata command.

PSObject

A metadata map object can be piped to the Remove-ProvTaskMetadata command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvTaskMetadata
- Set-ProvTaskMetadata
- Get-ProvTask
- Remove-ProvTask
- Stop-ProvTask
- Switch-ProvTask

Remove-ProvVM

March 11, 2024

Removes virtual machines created by Machine Creation Services.

Syntax

```
Remove-ProvVM
       [-ProvisioningSchemeName] <String>
       -VMName <String[]>
       [-PurgeDBOnly]
       [-ForgetVM]
       [-RunAsynchronously]
       [-PurgeJobOnSuccess]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Remove-ProvVM
       [-ProvisioningSchemeUid] <Guid>
       -VMName <String[]>
       [-PurgeDBOnly]
       [-ForgetVM]
       [-RunAsynchronously]
       [-PurgeJobOnSuccess]
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Allows you to remove VMs from the Machine Creation Services and hypervisor.

Examples

EXAMPLE 1

Remove all VMs from provisioning scheme XenPS

```
,(Get-ProvVM -ProvisioningSchemeName XenPS) | Remove-ProvVM XenPS

            TaskId                          : cfb506a5-cc7e-4a49-ac7b-
                dd960029d0d3
            Active                          : False
            Host                            : DDC
            DateStarted                     : 10/10/2021 16:39:45
            Metadata                        : {
 }

            CurrentOperation                :
            TaskProgress                    : 100
            LastUpdatedTime                 : 10/10/2021 16:39:50
            ActiveElapsedTime               : 5
            DateFinished                    : 10/10/2021 16:39:50
            TerminatingError                :
            Type                            : RemoveVirtualMachine
            Status                          : Finished
            PurgeDBOnly                     : False
            ForgetVM                        : False
            RemovedVirtualMachines          : {
XD\IP0001$, XD\IP0002$ }

            FailedVirtualMachines           : {
 }

            ProvisioningSchemeName          : XenPS
            ProvisioningSchemeUid           : e1afc8fb-3f52-42ea-9a17-305
                fdb0b6ee4
            TaskState                       : Finished
            TaskStateInformation            :
            WorkflowStatus                  : Completed
            TaskExpectedCompletion          : 10/10/2021 16:39:50
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme from which VMs will be removed.

Type:	String
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme from which VMs will be removed.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMName

List of VM names that will be removed from the provisioning scheme.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-PurgeDBOnly

If this option is specified, this command will only remove VM objects from the Machine Creation Services database; however, the VMs and hard disk copies still remain in the hypervisor. The hypervisor administrator can remove the VMs and hard disk images using the tools provided by the hypervisor

itself. If not specified, the VMs and hard disk copies are also removed from hypervisor storage. This parameter is exclusive with "ForgetVM".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ForgetVM

If this option is specified, this command will only remove VM objects from the Machine Creation Services database; however, the VMs and hard disk copies still remain in the hypervisor after removing VMs' citrix tags/identifiers. The hypervisor administrator can remove the VMs and hard disk images using the tools provided by the hypervisor itself. If not specified, the VMs and hard disk copies are also removed from hypervisor storage. This parameter is only applied to persistent VMs and exclusive with "PurgeDBOnly".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If this is specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named

Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMName'and 'ProvisioningSchemeName'to Remove-ProvVM.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored for the task. For new tasks, this is empty until metadata is added.

CurrentOperation <string>

Operation specific phase of the overall "Running"task state.

TaskProgress <double>

The progress of the task 0-100%.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For new remove-VM tasks, this is always "RemoveVirtualMachine".

Status <string>

Where in its lifecycle the task is.

PurgeDBOnly <bool>

Whether the named VMs are only removed from the Machine Creation Service database, but still remain in the hypervisor.

ForgetVm <bool>

Whether the named VMs are only removed from the Machine Creation Service database, but still remain in the hypervisor after removing VMs'citrix tags/identifiers.

RemovedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

Object array of removed virtual machines summary information. The object properties are:

ADAccountName <string>

The domain-qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents.

Message <string>

The exception message.

Details <string>

The full exception content including stack trace.

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state.

Status <string>

The status of the virtual machine.

StatusAdditionalInformation <string>

Extra information about the Status.

VMId <string>

The virtual machine identifier in the hypervisor.

VMName <string>

The display name of the virtual machine in the hypervisor.

FailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

See RemovedVirtualMachines for details of the object parameters.

ProvisioningSchemeName <string>

The name of the provisioning scheme from which the VM was created.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme from which the VM was created.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,

Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

Job failed for reasons specified in TaskStateInformation.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>

Provides more detailed information about the current task state.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

Notes

The cmdlet is associated with a task of type RemoveVirtualMachine, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

RemovingVirtualMachines

Only one long-running task for each provisioning scheme can be processed at a time.

This task still operates if the hosting unit or VMs in the hypervisor are missing. This removes the data from the Citrix Machine Creation Services, but the VMs remain in the hypervisor.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

InvalidParameterCombination

Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- New-ProvVM

Rename-ProvImageDefinition

March 11, 2024

Changes the parameter values for a image definition.

Syntax

```
Rename-ProvImageDefinition
      [-PreparedImageDefinitionName] <String>
      -NewPreparedImageDefinitionName <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-ProvImageDefinition
      -PreparedImageDefinitionUid <Guid>
      -NewPreparedImageDefinitionName <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to rename an existing image definition.

Examples

EXAMPLE 1

Renames an image definition from “CurrentName”to “NewName”.

```
Rename-ProvImageDefinition -PreparedImageDefinitionName CurrentName -
    NewPreparedImageDefinitionName NewName

CreateTime                      : 1/1/2022 12:00:00
CustomProperties                :
Description                     : Windows image
HostingUnitName                 : Hu
HostingUnitUid                  : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
OsType                          : Windows
PreparedImageDefinitionName     : NewName
PreparedImageDefinitionUid      : 7585f0de-192e-4847-a6d8-22713c3a2f42
UseWriteBackCache               : False
```

Parameters

-PreparedImageDefinitionName

The current name of the image definition.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionUid

The unique identifier of the image definition that is to be renamed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewPreparedImageDefinitionName

The new name for the image definition. This must not be a name that is being used by an existing image definition, and it must not contain any of the following characters \/;:#.*?=<>|[]()'""

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageDefinition

This object provides details of the image definitions and contains the following information:

CreateTime <DateTime>

The date and time when the image definition was created.

CustomProperties <string>

Properties of the image definition which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

Description <string>

The description for this image definition.

HostingUnitName <string>

The name of the hosting unit being used by this image definition.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this image definition.

OsType <string>

The OS type for this image definition.

PreparedImageDefinitionName <string>

The name of this image definition.

PreparedImageDefinitionUid <Guid>

The unique identifier for this image definition.

UseWriteBackCache <bool>

Whether to use WriteBackCache of this image definition.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageDefinition
- Remove-ProvImageDefinition
- New-ProvImageVersion
- Set-ProvImageVersion

Rename-ProvScheme

March 11, 2024

Renames a provisioning scheme.

Syntax

```
Rename-ProvScheme
      [-ProvisioningSchemeName] <String>
      [-NewProvisioningSchemeName] <String>
      [-PassThru]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Rename-ProvScheme
      -ProvisioningSchemeUid <Guid>
      [-NewProvisioningSchemeName] <String>
      [-PassThru]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to change the name of an existing provisioning scheme. The unique identifier of the provisioning scheme never changes after it has been created, therefore, the provisioning scheme can always be identified by its unique identifier regardless of any name changes.

Examples

EXAMPLE 1

Renames a provisioning scheme from “currentName”to “NewName”.

```
Rename-ProvScheme -provisioningSchemeName CurrentName -
    NewProvisioningSchemeName NewName
```

EXAMPLE 2

Renames a provisioning scheme from “currentName”to “NewName”and displays the resulting state.

```
Rename-ProvScheme -provisioningSchemeName CurrentName -
    NewProvisioningSchemeName NewName -PassThru

CleanOnBoot                  : True
ControllerAddress            : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }

CpuCount                     : 1
DiskSize                     : 20
HostingUnitName              : XenHU
HostingUnitUid               : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName             : idPool1
IdentityPoolUid              : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                 : 0
MachineProfile               :
MasterImageVM                : Base.vm
MasterImageVMDate            : 17/05/2020 09:53:40
MemoryMB                     : 1024
Metadata                     : {
 Department = Sales }

MetadataMap                  : {
 [Department = Sales] }

ProvisioningSchemeName       : NewName
ProvisioningSchemeUid        : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeVersion    : 1
TaskId                       : 00000000-0000-0000-0000-000000000000
VMMetadata                   : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter     :
PersonalVDiskDriveSize       : 0
UsePersonalVDiskStorage      : False
NetworkMaps                  : {
 0 }
```



```
36
37 Scopes                        :
38 DedicatedTenancy              : False
39 GpuTypeId                     :
40 ResetAdministratorPasswords   : False
41 SecurityGroups                : {
42   }
43
44 ServiceOffering               :
45 TenancyType                   : Shared
46 AzureAdJoinType               :
47 CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
48 CustomProperties              :
49 IdentityType:                 : ActiveDirectory
50 UseFullDiskCloneProvisioning  : False
51 UseWriteBackCache             : True
52 WriteBackCacheDiskSize        : 24
53 WriteBackCacheMemorySize      : 256
54 Warnings                      : {
55   }
56
57 WriteBackCacheDiskIndex       : 0
```

Parameters

-ProvisioningSchemeName

The current name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme that is to be renamed.

Type:	Guid
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NewProvisioningSchemeName

The new name for the provisioning scheme. This must not be a name that is being used by an existing provisioning scheme, and it must not contain any of the following characters \/;:#.*?=<>|[]()’””

Type:	String
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Defines whether the command returns a result showing the new state of the updated identity pool.

Type:	SwitchParameter
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

CleanOnBoot <bool>

Indicates whether the VMs created will be reset to a clean state on each start.

ControllerAddress <string[]>

The DNS names of the controllers associated with this provisioning scheme.

CpuCount <int>

The number of processors that will be used to create VMs.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

HostingUnitName <string>

The name of the hosting unit being used by this provisioning scheme.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this provisioning scheme.

IdentityPoolName <string>

The name of the identity pool being used by this provisioning scheme.

IdentityPoolUid <Guid>

The unique identifier of the identity pool being used by this provisioning scheme.

MachineCount <int>

The count of machines created with this provisioning scheme.

MachineProfile <string>

The inventory path to the source VM used by the provisioning scheme as a template.

MasterImageVM <string>

The inventory path to the VM snapshot copy used by this provisioning scheme.

MasterImageVMDate <DateTime>

The date and time when the VM snapshot copy used by this provisioning scheme was made.

MemoryMB <int>

The maximum amount of memory that will be used to created VMs in MB.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The metadata associated with this provisioning scheme.

MetadataMap <IDictionary[string, string];>

The metadata associated with this provisioning scheme arranged in key value pairs.

ProvisioningSchemeName <string>

The name of this provisioning scheme.

ProvisioningSchemeUid <Guid>

The unique identifier for this provisioning scheme.

ProvisioningSchemeVersion <int>

The version of the provisioning scheme.

TaskId <Guid>

The identifier of any current task that is running for the provisioning scheme.

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

PersonalVDiskDriveLetter <char>

The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int>

The size of the personal vDisk in GB.

UsePersonalVDiskStorage <bool>

Indicates whether this provisioning scheme uses personal vDisk storage.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]>

The NIC/network mappings that will be used to create VMs.

Scope <Citrix.MachineCreation.Sdk.ScopeReference[]>

The administration scopes associated with this provisioning scheme.

DedicatedTenancy <bool>

Indicates whether dedicated tenancy is used when creating VMs in Cloud Hypervisors.

GpuTypeId <string>

The id of the GPU (Graphics Processor Unit) Type used by this scheme. It is null if there is no GPU.

ResetAdministratorPasswords <bool>

Indicates whether the passwords for administrator accounts are reset on created machines.

ServiceOffering <string>

The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]>

The security groups that will be applied to machines created in Cloud Hypervisors.

TenancyType <string>

Tenancy type to be used when creating VMs in Cloud Hypervisors. (See New-ProvScheme.)

AzureAdJoinType <string>

Deprecated.

CurrentMasterImageUid <Guid>

The unique identifier of the current master image used by the provisioning scheme. (See Get-ProvSchemeMasterVMImageHistory.)

CustomProperties <string>

Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

IdentityType <string>

Identity type used to join created machines to a directory service. (See New-ProvScheme.)

UseFullDiskCloneProvisioning <bool>

Indicates whether VMs will be created using the dedicated full disk clone feature.

UseWriteBackCache <bool>

Indicates whether this provisioning scheme will use the write-back cache feature.

WriteBackCacheDiskSize <int>

The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int>

The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNameAlreadyExists

A provisioning scheme with the same name as the new provisioning scheme name already exists.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvScheme
- Set-ProvScheme
- Get-ProvScheme
- Test-ProvSchemeNameAvailable

Request-ProvVMUpdate

March 11, 2024

(DEPRECATED) Requests a property update on provisioned virtual machines.

Syntax

```
Request-ProvVMUpdate
       -ProvisioningSchemeName <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Request-ProvVMUpdate
       -ProvisioningSchemeName <String>
       -VMName <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Request-ProvVMUpdate
       -ProvisioningSchemeUid <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Request-ProvVMUpdate
       -ProvisioningSchemeUid <Guid>
       -VMName <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

This cmdlet has been deprecated and replaced by Set-ProvVMUpdateTimeWindow. Please see the examples on this page that describe how to transition to the new cmdlet.

Provides the ability to synchronize the properties of existing virtual machines with any changes occurred on the provisioning scheme.

This cmdlet updates the ProvisioningSchemeUpdateRequested field on virtual machines to the current time, which indicates that the machines will synchronize to the property updates occurred on the provisioning scheme upon next start.

Examples

EXAMPLE 1

To request all machines in a catalog to apply property updates on next boot, use Set-ProvVMUpdateTimeWindow with parameters -StartsNow and -DurationInMinutes -1.

```
Request-ProvVMUpdate -ProvisioningSchemeName AzuPS

is equivalent to

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName AzuPS -StartsNow -
    DurationInMinutes -1
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

List of VM names requesting for update. If not specified, all VMs in the machine catalog associated with the specified provisioning scheme will request for a property update.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId'and 'ProvisioningSchemeName'to Request-ProvVMUpdate.

Outputs

None

By default, this cmdlet returns no output.

Notes

Running Set-ProvScheme to update provisioning scheme properties will only affect future provisioned machines. To request existing VMs in a catalog to be brought up to date upon the next power on, use this cmdlet to request certain or all VMs created from the provisioning scheme associated with the catalog to be updated.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvVMUpdateTimeWindow

Reset-ProvEnabledFeatureList

March 11, 2024

Refreshes the MachineCreation service's list of enabled features.

Syntax

```
Reset-ProvEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix MachineCreation Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected MachineCreation service instance's list of enabled features.

```
Reset-ProvEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-ProvServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the MachineCreation Service.

Syntax

```
Reset-ProvServiceGroupMembership
     [-ConfigServiceInstance] <ServiceInstance[]>
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Enables you to reload MachineCreation Service access permissions and configuration service locations. The Reset-ProvServiceGroupMembership command must be run on at least one instance of the service type (Prov) after installation and registration with the configuration service. Without this operation, the MachineCreation services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-ProvServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    ProvServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-ProvServiceGroupmembership
```


Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-ProvServiceGroupMembership command.

Outputs

Citrix.MachineCreation.Sdk.ServiceInstance

Reset-ProvServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the MachineCreation Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvServiceInstance
- Get-ProvServiceStatus

Reset-ProvVMDisk

March 11, 2024

Resets the OS disk of persistent VMs or the Identity disk of persistent and non-persistent VMs.

Syntax

```
Reset-ProvVMDisk
      [-ProvisioningSchemeName] <String>
      -VMName <String[]>
      [-RunAsynchronously]
      [-OS]
      [-PurgeJobOnSuccess]
      [-Force]
      [-LoggingId <Guid>]
      [-WhatIf]
      [-Confirm]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Reset-ProvVMDisk
     [-ProvisioningSchemeName] <String>
     -VMName <String[]>
     [-RunAsynchronously]
     [-Identity]
     [-ResetIdentityInfo]
     [-PurgeJobOnSuccess]
     [-Force]
     [-LoggingId <Guid>]
     [-WhatIf]
     [-Confirm]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Reset-ProvVMDisk
     [-ProvisioningSchemeUid] <Guid>
     -VMName <String[]>
     [-RunAsynchronously]
     [-OS]
     [-PurgeJobOnSuccess]
     [-Force]
     [-LoggingId <Guid>]
     [-WhatIf]
     [-Confirm]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

```
Reset-ProvVMDisk
     [-ProvisioningSchemeUid] <Guid>
     -VMName <String[]>
     [-RunAsynchronously]
     [-Identity]
     [-ResetIdentityInfo]
     [-PurgeJobOnSuccess]
     [-Force]
     [-LoggingId <Guid>]
     [-WhatIf]
     [-Confirm]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Allows you to reset the disks of persistent VMs from Machine Creation Services and hypervisor.

Examples

EXAMPLE 1

Reset the OS disks of all VMs in provisioning scheme myProvScheme

```
,(Get-ProvVM -ProvisioningSchemeName "myProvScheme") | Reset-ProvVMDisk
    -ProvisioningSchemeName "myProvScheme" -OS


TaskId                  : 2706e50b-aff9-44c7-a3e3-8c8c7f1f4ac5
Active                  : False
Host                    : MY-DDC
DateStarted             : 5/24/2022 12:02:29 PM
Metadata                : {
  }

CurrentOperation        :
TaskProgress            : 100
LastUpdateTime          : 5/24/2022 12:02:41 PM
ActiveElapsedTime       : 10
DateFinished            : 5/24/2022 12:02:41 PM
TerminatingError        :
Type                    : ResetVM
Status                  : Finished
ResetVirtualMachines    : {
 XD\machine01$, XD\machine02$ }

FailedVirtualMachines   : {
  }

ProvisioningSchemeName : myProvScheme
ProvisioningSchemeUid  : 3d79b132-22c5-40d3-9a99-7cb497924f9e
TaskState               : Finished
TaskStateInformation    :
WorkflowStatus          : Completed
TaskExpectedCompletion :
```

EXAMPLE 2

Reset the OS disks of machine01 and machine02 in provisioning scheme myProvScheme

```
Reset-ProvVMDisk -ProvisioningSchemeName "myProvScheme" -VMName ("
    machine01","machine02") -OS


TaskId                  : 2706e50b-aff9-44c7-a3e3-8c8c7f1f4ac5
Active                  : False
Host                    : MY-DDC
DateStarted             : 5/24/2022 12:02:29 PM
Metadata                : {
```



```
  }

CurrentOperation       :
TaskProgress           : 100
LastUpdateTime         : 5/24/2022 12:02:41 PM
ActiveElapsedTime      : 10
DateFinished           : 5/24/2022 12:02:41 PM
TerminatingError       :
Type                   : ResetVM
Status                 : Finished
ResetVirtualMachines   : {
 XD\machine01$, XD\machine02$ }

FailedVirtualMachines  : {
  }

ProvisioningSchemeName : myProvScheme
ProvisioningSchemeUid  : 3d79b132-22c5-40d3-9a99-7cb497924f9e
TaskState              : Finished
TaskStateInformation   :
WorkflowStatus         : Completed
TaskExpectedCompletion :
```

EXAMPLE 3

Reset the OS disk of machine01 in provisioning scheme myProvScheme

```
Reset-ProvVMDisk -ProvisioningSchemeName "myProvScheme" -VMName "
    machine01" -OS


TaskId                 : 2706e50b-aff9-44c7-a3e3-8c8c7f1f4ac5
Active                 : False
Host                   : MY-DDC
DateStarted            : 5/24/2022 12:02:29 PM
Metadata               : {
  }

CurrentOperation       :
TaskProgress           : 100
LastUpdateTime         : 5/24/2022 12:02:41 PM
ActiveElapsedTime      : 10
DateFinished           : 5/24/2022 12:02:41 PM
TerminatingError       :
Type                   : ResetVM
Status                 : Finished
ResetVirtualMachines   : {
 XD\machine01$ }

FailedVirtualMachines  : {
  }

```



```
ProvisioningSchemeName : myProvScheme
ProvisioningSchemeUid  : 3d79b132-22c5-40d3-9a99-7cb497924f9e
TaskState              : Finished
TaskStateInformation   :
WorkflowStatus         : Completed
TaskExpectedCompletion :
```

EXAMPLE 4

Reset all configurations in the identity disks of all VMs in provisioning scheme myProvScheme

```
,(Get-ProvVM -ProvisioningSchemeName "myProvScheme") | Reset-ProvVMDisk
    -ProvisioningSchemeName "myProvScheme" -Identity
```

EXAMPLE 5

Only reset the machine password and trust keys in the configuration file in the identity disk of VM machine01

```
Reset-ProvVMDisk -ProvisioningSchemeName "myProvScheme" -VMName "
   machine01" -Identity -ResetIdentityInfo
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme from which VM disks will be reset. This feature is available on VMware, Azure, AWS and Citrix hypervisor.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme from which VM disks will be reset.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-OS

Indicates that the OS disk should be reset. This parameter is exclusive with "Identity". Either "OS"or "Identity"must be provided.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Identity

Indicates that the identity disk should be reset. This parameter is exclusive with "OS". "Repair-AcctIdentity -Target IdentityInfo"should be executed before this command. By default, all configurations in the identity disk will be reset except the DeviceKeyPair of hybrid Azure AD (if feature applied). If DeviceKeyPair is desired to be reset, Set-AcctADAccountUserCert should be executed in advance.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMName

List of VM names that will be have the disk reset.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ResetIdentityInfo

Indicates whether only machine password and trust keys in the configuration file will be reset. This parameter is only used with "Identity".

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether the command returns before it completes. If this is specified, the command returns an identifier for the task that was created. This task can be monitored using the Get-ProvTask command.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history is removed from the database when the task completes. This cannot be specified for tasks that run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

Suppresses prompts for confirmation before proceeding to reset the disk of the specified VMs.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WhatIf

Shows what would happen if the cmdlet runs. The cmdlet is not run.

Type:	SwitchParameter
Aliases:	wi
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Confirm

Prompts for confirmation before running the cmdlet.

Type:	SwitchParameter
Aliases:	cf
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMName'and 'ProvisioningSchemeName'to Reset-ProvVMDisk.

Outputs

Guid

When the RunAsynchronously identifier is specified, this GUID is returned and provides the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task that was run and contains the following information: TaskId <Guid>

The identifier for the task that was performed.

Active <bool>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time when the task was started.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored for the task. For new tasks, this is empty until metadata is added.

CurrentOperation <string>

Operation specific phase of the overall “Running”task state.

TaskProgress <double>

The progress of the task 0-100%.

LastUpdateTime <DateTime>

The date and time of the last task status update.

ActiveElapsedTime <int>

Number of seconds the task has taken for active execution.

DateFinished <DateTime>

The date and time when the task was completed.

TerminatingError < Citrix.Fma.Sdk.ServiceCore.CommonCmdlets.TaskterminatingError>

Diagnostic information if the task completely fails.

Type <Citrix.XDInterServiceTypes.JobType>

The type of task. For new reset disk tasks, this is always “Reset-ProvVMDisk”.

Status <string>

Where in its lifecycle the task is.

ResetVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

See FailedVirtualMachines for details of the object parameters.

FailedVirtualMachines <Citrix.DesktopUpdateManager.SDK.VirtualMachineCreation[]>

ADAccountName <string>

The domain-qualified AD Account name of the machine.

ADAccountSid <string>

The AD account SID of the machine account.

DiagnosticInformation <Citrix.MachineCreation.Sdk.ExceptionSurrogate[]>

A collection of handled error states which caused the provisioning to fail.

ExceptionType <string>

The type of exception this object represents.

Message <string>

The exception message.

Details <string>

The full exception content including stack trace.

InnerException <Citrix.MachineCreation.Sdk.ExceptionSurrogate>

Information relating to any contributing error state.

Status <string>

The status of the virtual machine.

StatusAdditionalInformation <string>

Extra information about the Status.

VMId <string>

The virtual machine identifier in the hypervisor.

VMName <string>

The display name of the virtual machine in the hypervisor.

ProvisioningSchemeName <string>

The name of the provisioning scheme from which the VM was created.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme from which the VM was created.

TaskState <Citrix.DesktopUpdateManager.SDK.ProvisionVMState>

The state of the task. This can be any of the following:

Processing

Indicates that the task is in its initial state.

LocatingResources,

Indicates that the task is locating information from other services.

HostingUnitNotFound

Indicates that the required hosting unit could not be located.

ProvisioningSchemeNotFound

Indicates that the required provisioning scheme could not be located.

Provisioning

Indicates that the task is at the provisioning stage.

IdentityPoolNotFound

Indicates that the required identity pool could not be located.

TaskAlreadyRunningForProvisioningScheme

Indicates that the provisioning scheme already has another task running.

Finalizing

Indicates that the task is finalizing.

Finished

Indicates that the task is complete.

FinishedWithErrors

Indicates that the task is complete but there were errors. Specific details of errors are included with each failed virtual machine.

Removing

Indicates that the task is removing virtual machines from the hypervisor.

Failed

Job failed for reasons specified in TaskStateInformation.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask)

TaskStateInformation <string>

Provides more detailed information about the current task state.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is used to process the task.

TaskExpectedCompletion <DateTime>

The date and time at which the task is expected to complete.

Notes

Only one long-running task for each provisioning scheme can be processed at a time.

In the case of failure, the following errors can result.

Error Codes ————ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation. Communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

InvalidParameterCombination

Both PurgeJobOnSuccess and RunAsynchronously were specified. When running asynchronously, the cmdlet terminates before the job does, so it cannot clean up the completed job.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

The cmdlet is associated with a task of type ResetVirtualMachine, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

ResettingVirtualMachines

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- New-ProvVM
- Repair-AcctIdentity

Schedule-ProvVMUpdate

March 11, 2024

(DEPRECATED) Schedules a property update on provisioned virtual machines.

Syntax

```
Schedule-ProvVMUpdate
        [-StartTimeInUTC <String>]
        [-StartsNow]
        [-DurationInMinutes <Int32>]
        -ProvisioningSchemeName <String>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Schedule-ProvVMUpdate
        [-StartTimeInUTC <String>]
        [-StartsNow]
        [-DurationInMinutes <Int32>]
        -ProvisioningSchemeName <String>
        -VMName <String[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Schedule-ProvVMUpdate
        [-StartTimeInUTC <String>]
        [-StartsNow]
        [-DurationInMinutes <Int32>]
        -ProvisioningSchemeUid <Guid>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

```
Schedule-ProvVMUpdate
        [-StartTimeInUTC <String>]
```



```
        [-StartsNow]
        [-DurationInMinutes <Int32>]
        -ProvisioningSchemeUid <Guid>
        -VMName <String[]>
        [-LoggingId <Guid>]
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

This cmdlet has been deprecated and replaced by %5BSet-ProvVMUpdateTimeWindow%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Set-ProvVMUpdateTimeWindow.html). %5BSet-ProvVMUpdateTimeWindow%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Set-ProvVMUpdateTimeWindow.html) is the new name for Schedule-ProvVMUpdate and provides all the same functionality, please use it going foward in place of Schedule-ProvVMUpdate.

Provides the ability to synchronize the properties of existing virtual machines with any changes occurred on the provisioning scheme.

This cmdlet defines a time window where a provisioning scheme update will take place against either specific machines

in a catalog or an entire catalog. The update will take place when the machine is next booted within the time window.

On each VM an update is scheduled, ProvisioningSchemeUpdateRequested defines the start of the time window and

ProvisioningSchemeUpdateUntil defines the end of the time window. These fields will be cleared upon a successful

provisioning scheme update or if the time window has expired when the machine has booted

Examples

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

List of VM names to schedule an update for. If not specified, all VMs in the machine catalog associated with the specified provisioning scheme will be scheduled for a property update

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme

Type:	Guid
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTimeInUTC

The start time in UTC format of the time window the schedule is active for. Mutually exclusive with StartsNow switch. Must be no more than 5 seconds in the past. The string will be converted into a DateTime using parsing rules defined by the DateTime.Parse method, which allows for a timezone offset to specify time in any timezone. If no timezone offset is specified, UTC will be assumed. See the example usages of Schedule-ProvVMupdate or this link for more details https://docs.microsoft.com/en-us/dotnet/api/system.datetime.parse?view=net-6.0#the-string-to-parse

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartsNow

Whether this schedule should start right away. Mutually exclusive with specific StartTimeInUTC. Implied to be set if StartTimeInUTC is not specified

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DurationInMinutes

The duration in minutes of the time window the schedule is active for. If less than 0, indicates the schedule has no end time

Type:	Int32
Position:	Named
Default value:	120 minutes
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId'and 'ProvisioningSchemeName'to Schedule-ProvVMUpdate.

Outputs

None

By default, this cmdlet returns no output.

Notes

Running Set-ProvScheme to update provisioning scheme properties will only affect future provisioned machines. To schedule existing VMs in a catalog to be brought up to date upon the next power on, use this cmdlet to schedule certain or all VMs created from the provisioning scheme associated with the catalog to be updated.

Related Links

- about_ProvMachineCreationSnapin
- Set-ProvVMUpdateTimeWindow

Set-ProvDBConnection

March 11, 2024

Configures a database connection for the MachineCreation Service.

Syntax

```
Set-ProvDBConnection
    [-DBConnection] <String>
    [-Force]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Specifies the database connection string for use by the currently selected Citrix MachineCreation Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
1  Set-ProvDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
       Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
1  Set-ProvDBConnection -DBConnection $null
```


Parameters

-DBConnection

Specifies the database connection string to be used by the MachineCreation Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-ProvDBConnection cmdlet returns an object describing the status of the MachineCreation Service together with extra diagnostics information. Possible values are:

- OK:

The MachineCreation Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the MachineCreation Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the MachineCreation Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the MachineCreation Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the MachineCreation Service currently in use is newer than, and incompatible with, the version of the MachineCreation Service schema on the database. Upgrade the MachineCreation Service to a more recent version.

- DBOlderVersionThanService:

The version of the MachineCreation Service schema on the database is newer than, and incompatible with, the version of the MachineCreation Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the MachineCreation Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the MachineCreation Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the MachineCreation Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvServiceStatus
- Get-ProvDBConnection
- Test-ProvDBConnection

Set-ProvDBCredentials

March 11, 2024

Configures the database server SQL credentials for the MachineCreation Service.

Syntax

```
Set-ProvDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix MachineCreation Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional MachineCreation Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-ProvDBCredentials
```


EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-ProvDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-ProvDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-ProvDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-ProvDBSchema
- Set-ProvDBConnection
- Get-Credential

Set-ProvImageDefinition

March 11, 2024

Changes the parameter values for a image definition.

Syntax

```
Set-ProvImageDefinition
   [-PreparedImageDefinitionUid] <Guid>
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageDefinition
   -PreparedImageDefinitionName <String>
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to update the parameters of an existing image definition.

The following parameters are allowed to be updated:

- The description of the image definition.

Examples

EXAMPLE 1

Update a image definition name with uid “7585f0de-192e-4847-a6d8-22713c3a2f42”.

```
Set-ProvImageDefinition -PreparedImageDefinitionUid 7585f0de-192e-4847-
   a6d8-22713c3a2f42 -Description 'Windows image'

CreateTime                    : 1/1/2022 12:00:00
CustomProperties              :
Description                   : Windows image
HostingUnitName               : Hu
HostingUnitUid                : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
OsType                        : Windows
PreparedImageDefinitionName   : MyImage
PreparedImageDefinitionUid    : 7585f0de-192e-4847-a6d8-22713c3a2f42
UseWriteBackCache             : False
```


Parameters

-PreparedImageDefinitionUid

The identifier of the image definition to be updated.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name of the image definition to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

A value indicating the image definition description.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageDefinition

This object provides details of the image definitions and contains the following information:

CreateTime <DateTime>

The date and time when the image definition was created.

CustomProperties <string>

Properties of the image definition which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

Description <string>

The description for this image definition.

HostingUnitName <string>

The name of the hosting unit being used by this image definition.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this image definition.

OsType <string>

The OS type for this image definition.

PreparedImageDefinitionName <string>

The name of this image definition.

PreparedImageDefinitionUid <Guid>

The unique identifier for this image definition.

UseWriteBackCache <bool>

Whether to use WriteBackCache of this image definition.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageDefinition
- New-ProvImageDefinition
- Remove-ProvImageDefinition
- Rename-ProvImageDefinition

Set-ProvImageVersion

March 11, 2024

Changes the parameter values for a image version.

Syntax

```
Set-ProvImageVersion
   [-PreparedImageVersionUid] <Guid>
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageVersion
   -PreparedImageDefinitionName <String>
   -PreparedImageVersionNumber <String>
   [-Description <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to update the parameters of an existing image version.

The following parameters are allowed to be updated:

- The description of the image version.

Examples

EXAMPLE 1

Update a image definition description with image definition name “MyImage”and image version number “1”.

```
Set-ProvImageVersion -PreparedImageDefinitionName MyImage -
   PreparedImageVersionNumber 1 -Description "Windows image"

CreateTime                        : 5/19/2022 3:20:23 AM
Description                       : Windows image
DiskSize                          : 32
Error                             :
HostingUnitUid                    : b86b4442-6ecb-4b73-a655-7ca7170e19e5
ImageRuntimeInfo                  :
ImageStatus                       : Prepared
ImageVersionMetadata              : {
  }

MasterImageId                     : hu-dev-testing-rg/hu-dev-tsvda-
   snapshot
```



```
MasterImageVM                    : /East US.region/image.folder/hu-dev-
    testing-rg.resourcegroup/hu-dev-tsvda-snapshot.snapshot
PageFileSettings                 : D:\pagefile.sys 0 0
PreparedImageDefinitionName      : MyImage
PreparedImageDefinitionUid       : 6d67c0a5-6ecc-4c46-8404-156205635cf4
PreparedImageReplicas            : {
 57fc60c3-eb9b-4d38-8646-0afceec85335 }

PreparedImageSchemeUid           : 57ce7bbc-7193-47df-95d4-d4999c720cac
PreparedImageVersionNumber       : 2
PreparedImageVersionUid          : d366d140-1ec0-464f-aea0-77b706e94580
UpdatePageFileSettings           : False
VMMetadata                       :
Warnings                         : {
  }

WriteBackCacheDiskIndex          : 0
WriteBackCacheDiskSize           : 0
WriteBackCacheMemorySize         : 0
```

Parameters

-PreparedImageVersionUid

The identifier of the image version to be updated.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name of the image definition to be updated.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionNumber

The version number of the image version to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

The description of the image version to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ImageDefinition

This object provides details of the image definitions and contains the following information:

CreateTime <DateTime>

The date and time when the image version was created.

Description <string>

The description of the image version.

DiskSize <int>

The disk size (in GB) that will be used to create VMs.

Error <string>

Error state that has occurred with this image version.

HostingUnitUid <Guid>

The unique identifier of the hosting unit being used by this image scheme.

ImageRuntimeInfo <string>

The image runtime information like operating system info, VDA components, etc.

ImageStatus <string>

The status of the provisioning scheme image.

ImageVersionMetadata <string>

The metadata associated with this image version.

MasterImageId <string>

The unique identifier of the snapshot, as assigned by the hosting unit.

MasterImageVM <string>

The inventory path to the VM snapshot copy used by this provisioning scheme.

PageFileSettings <string>

The page file settings to meet hypervisor-specific features demands.

PreparedImageDefinitionName <string>

The name of the image definition which is used for this image version.

PreparedImageDefinitionUid <Guid>

The unique identifier of the image definition which is used for this image version.

PreparedImageReplicas

Image replication information for this image version.

PreparedImageSchemeUid <Guid>

The unique identifier of the image scheme which is used for this image version.

PreparedImageVersionNumber <string>

The version number of the image version.

PreparedImageVersionUid <Guid>

The unique identifier for the image version.

UpdatePageFileSettings <bool>

Whether to update page file settings for write back cache

VMMetadata <char[]>

The metadata that will be used to created VMs in a plain text format.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]>

Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int>

The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

WriteBackCacheDiskSize <int>

The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int>

The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

UnsupportedByServer

The requested operation is not supported by this version of the service.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvImageVersion
- New-ProvImageVersion
- Remove-ProvImageVersion

Set-ProvImageVersionMetadata

March 11, 2024

Adds or updates metadata on the given image version.

Syntax

```
Set-ProvImageVersionMetadata
   [-PreparedImageVersionUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageVersionMetadata
   [-PreparedImageVersionUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ProvImageVersionMetadata
   [-PreparedImageDefinitionName] <String>
   [-PreparedImageVersionNumber] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageVersionMetadata
   [-PreparedImageDefinitionName] <String>
   [-PreparedImageVersionNumber] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageVersionMetadata
   [-InputObject] <ImageVersion[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvImageVersionMetadata
   [-InputObject] <ImageVersion[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given image version objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the image version with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ProvImageVersionMetadata -PreparedImageVersionUid 4CECC26E-48E1-423
   F-A1F0-2A06DDD0805C -Name property -Value value

Key                                     Value
```



```
---                                     -----
property                                value
```

Parameters

-PreparedImageVersionUid

Id of the image version

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-PreparedImageDefinitionName

Name of the image definition

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-PreparedImageVersionNumber

Version number of the image version

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

ImageVersion objects to which the metadata is to be added or updated.

Type:	ImageVersion[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added or updated. The property must be unique for the provisioning scheme specified. If the name already exists, its value is updated. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Aliases:	Property
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ImageVersion

You can pipe a ImageVersion object or any object containing a parameter called 'PreparedImageDefinitionName'and 'PreparedImageVersionNumber'or 'PreparedImageVersionUid'to Set-ProvImageVersionMetadata.

PSObject

A metadata map object can be piped to the Set-ProvImageVersionMetadata command.

Outputs

System.Collections.Generic.Dictionary[string, string]

Set-ProvImageVersionMetadata returns a dictionary containing the new (name, value)-pairs for the metadata being set.

Key <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes ————InvalidParameterCombination

The cmdlet parameters are inconsistent.

ObjectNotFound

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvImageVersionMetadata
- Remove-ProvImageVersionMetadata

Set-ProvResourceTags

March 11, 2024

Set provisioning resources tags.

Syntax

```
Set-ProvResourceTags
   [-ProvisioningSchemeName] <String>
   [-InputObject <ProvisionedVirtualMachine[]>]
   [-VMName <String>]
   [-VMBatchSize <Int32>]
   [-ResourceType <ResourceType>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvResourceTags
   -ProvisioningSchemeUid <Guid>
   [-InputObject <ProvisionedVirtualMachine[]>]
   [-VMName <String>]
   [-VMBatchSize <Int32>]
   [-ResourceType <ResourceType>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to update the tags of the resources created by Citrix during provisioning. The two types of resources created by Citrix are machine catalog level, and virtual machine level. They can be tagged separately or at the same time. Resources are tagged with 'CitrixProvisioningSchemeId', 'CitrixVirtualSiteId'and 'CitrixCustomerId'.

Examples

EXAMPLE 1

Tag MachineCatalog resources in provisioning scheme called "MyProvisioningSchemeName".

```
Set-ProvResourceTags -ProvisioningSchemeName MyProvisioningSchemeName -
    ResourceType MachineCatalog -OutVariable result

```



```
          SuccessfulVirtualMachineCount FailedVirtualMachineCount
             TagOperationErrors
          ----------------------------- ------------------------
             ------------------
                                      0                         0 {
 image.snapshot }



          $result.TagOperationErrors

          ResourceName   ResourceType   TagFailureReason
          ------------   ------------   ----------------
          image.snapshot MachineCatalog OperationNotAllowed
```

EXAMPLE 2

Tag both MachineCatalog and all VirtualMachine resources in provisioning scheme called “MyProvisioningSchemeName”.

```
,(Get-ProvVM -ProvisioningSchemeName myProvisioningSchemeName) | Set-
   ProvResourceTags -ProvisioningSchemeName MyProvisioningSchemeName -
   OutVariable result

          SuccessfulVirtualMachineCount FailedVirtualMachineCount
             TagOperationErrors
          ----------------------------- ------------------------
             ------------------
                                      5                         2 {
 myVirtualMachine0, myVirtualMachine1 }


          $result.TagOperationErrors

          ResourceName      ResourceType   TagFailureReason
          ------------      ------------   ----------------
          myVirtualMachine0 VirtualMachine OperationNotAllowed
          myVirtualMachine1 VirtualMachine OperationNotAllowed
```

EXAMPLE 3

Tag VirtualMachine MyVirtualMachine0’s resources in provisioning scheme called “MyProvisioningSchemeName”.

```
Set-ProvResourceTags -ProvisioningSchemeName MyProvisioningSchemeName -
   VMName MyVirtualMachine0 -ResourceType VirtualMachine -OutVariable
   result

```



```
SuccessfulVirtualMachineCount FailedVirtualMachineCount
   TagOperationErrors
---------------------------- ------------------------
   ------------------
                           1                        0 {
  }
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to be tagged.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme to be tagged.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InputObject

The VM name list to be tagged.

Type:	ProvisionedVirtualMachine[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

The VM name to be tagged.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMBatchSize

The maximum number of VMs to be tagged in a batch. Range 1-60 with default of 10.

Type:	Int32
Position:	Named
Default value:	10
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceType

Specifies the type of resource. Supported resource types are:

MachineCatalog

VirtualMachine

All

Type:	ResourceType
Position:	Named
Default value:	All
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Set-ProvResourceTags.

Outputs

Citrix.MachineCreation.Sdk.TagOperationDetailedSummary

The Set-ProvResourceTags that contains the following parameters:

SuccessfulVirtualMachineCount <int>

The number of virtual machines to which their machine level resources were tagged successfully.

FailedVirtualMachineCount <int>

The number of virtual machines to which their machine level resources were not tagged.

TagOperationErrors <Citrix.MachineCreation.Sdk.TagOperationError[]>

The list of resources that failed to be tagged. Each one has the following parameters:

ResourceType <string>

Type of the resource.

ResourceName <string>

Name of the resource.

TagFailureReason <string>

Reason for the tag operation failure. This can be one of the following:

OperationNotAllowed

The operation is not allowed because the resources are not in the right state to tag.

TooManyRequests

The request is being throttled because too many requestes arrive.

UnexpectedTagFailure

Failed to tag with unexpected exception.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

SetProvSchemeTagsFailed

An unexpected error occurred while setting machine catalog level resources tags.

SetProvVMTagsFailed

An unexpected error occurred while setting virtual machine level resources tags.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

Set-ProvScheme

March 11, 2024

Changes the parameter values for a provisioning scheme.

Syntax

```
Set-ProvScheme
    [-Version <Int32>]
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvScheme
    [-Version <Int32>]
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolName <String>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvScheme
    [-Version <Int32>]
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvScheme
    [-Version <Int32>]
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvScheme
    [-Version <Int32>]
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolName <String>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvScheme
    [-Version <Int32>]
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to update the parameters of an existing provisioning scheme. It increments the provisioning scheme's ProvisioningSchemeVersion by 1 each time it is called.

The following parameters are allowed to be updated:

- Number of CPUs that will be used for VMs created from the provisioning scheme.
- Maximum amount of memory that will be used for VMs created from the provisioning scheme.
- Identity pool that will be used for VMs created from the provisioning scheme.
- Machine profile that will be used for VMs created from the provisioning scheme.
- Cloud service offering that will be used for VMs created from the provisioning scheme.
- Custom properties of the provisioning scheme that are specific to the target hosting infrastructure.

To change the provisioning scheme name, see Rename-ProvScheme.

Examples

EXAMPLE 1

Updates a provisioning scheme called "MyScheme"to use two processors on the VMs that are created from the provisioning scheme.

```
Set-ProvScheme -ProvisioningSchemeName MyScheme -VMCpuCount 2


CleanOnBoot                 : True
ControllerAddress           : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }

CpuCount                    : 2
DiskSize                    : 20
HostingUnitName             : XenHU
HostingUnitUid              : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
IdentityPoolName            : idPool1
IdentityPoolUid             : 03743136-e43b-4a87-af74-ab71686b3c16
MachineCount                : 0
MachineProfile              :
MasterImageVM               : Base.vm
MasterImageVMDate           : 17/05/2020 09:53:40
MemoryMB                    : 1024
Metadata                    : {
```



```
 Department = Sales }

MetadataMap                    : {
 [Department = Sales] }

ProvisioningSchemeName         : MyScheme
ProvisioningSchemeUid          : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeVersion      : 1
TaskId                         : 00000000-0000-0000-0000-000000000000
VMMetadata                     : {
 0, 1, 0, 0... }

PersonalVDiskDriveLetter       :
PersonalVDiskDriveSize         : 0
UsePersonalVDiskStorage        : False
NetworkMaps                    : {
 0 }

Scopes                         :
DedicatedTenancy               : False
GpuTypeId                      :
ResetAdministratorPasswords    : False
SecurityGroups                 : {
  }

ServiceOffering                :
TenancyType                    : Shared
AzureAdJoinType                :
CurrentMasterImageUid          : c0571690-4f57-4476-901b-fe64d6aecb79
CustomProperties               :
IdentityType:                  : ActiveDirectory
UseFullDiskCloneProvisioning   : False
UseWriteBackCache              : True
WriteBackCacheDiskSize         : 24
WriteBackCacheMemorySize       : 256
Warnings                       : {
  }

WriteBackCacheDiskIndex        : 0
```

EXAMPLE 2

Updates a provisioning scheme called “MyScheme”to use the machine profile called 1.0.

```
Set-ProvScheme -ProvisioningSchemeName MyScheme -MachineProfile XDHyp:\
   HostingUnits\AzureHostingUnit\machineprofile.folder\RG.resourcegroup
   \TS.templatespec\1.0.templatespecversion


CleanOnBoot                    : True
ControllerAddress              : {
 ddcA.citrix.com,ddcB.citrix.com,ddcC.citrix2.com }
```



```
 7
 8  CpuCount                      :
 9  DiskSize                      : 20
10  HostingUnitName               : AzureHostingUnit
11  HostingUnitUid                : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
12  IdentityPoolName              : idPool1
13  IdentityPoolUid               : 03743136-e43b-4a87-af74-ab71686b3c16
14  MachineCount                  : 0
15  MachineProfile                : XDHyp:\HostingUnits\AzureHostingUnit\
       machineprofile.folder\RG.resourcegroup\TS.templatespec\1.0.
       templatespecversion
16  MasterImageVM                 : Base.vm
17  MasterImageVMDate             : 17/05/2020 09:53:40
18  MemoryMB                      : 1024
19  Metadata                      : {
20   Department = Sales }
21
22  MetadataMap                   : {
23   [Department = Sales] }
24
25  ProvisioningSchemeName        : MyScheme
26  ProvisioningSchemeUid         : 7585f0de-192e-4847-a6d8-22713c3a2f42
27  ProvisioningSchemeVersion     : 1
28  TaskId                        : 00000000-0000-0000-0000-000000000000
29  VMMetadata                    : {
30   0, 1, 0, 0... }
31
32  PersonalVDiskDriveLetter      :
33  PersonalVDiskDriveSize        : 0
34  UsePersonalVDiskStorage       : False
35  NetworkMaps                   : {
36   0 }
37
38  Scopes                        :
39  DedicatedTenancy              : False
40  GpuTypeId                     :
41  ResetAdministratorPasswords   : False
42  SecurityGroups                : {
43    }
44
45  ServiceOffering               :
46  TenancyType                   : Shared
47  AzureAdJoinType               :
48  CurrentMasterImageUid         : c0571690-4f57-4476-901b-fe64d6aecb79
49  CustomProperties              :
50  IdentityType:                 : ActiveDirectory
51  UseFullDiskCloneProvisioning  : False
52  UseWriteBackCache             : True
53  WriteBackCacheDiskSize        : 24
54  WriteBackCacheMemorySize      : 256
55  Warnings:                     : {
56    }
57
```



```
WriteBackCacheDiskIndex      : 0
```

EXAMPLE 3

Updates a provisioning scheme to have a new MachineProfile. The provisioning scheme state is then reverted back to version 1 and the original MachineProfile.

```
Get-ProvScheme | select ProvisioningSchemeName, MachineProfile,
    ProvisioningSchemeVersion

ProvisioningSchemeName MachineProfile

    ProvisioningSchemeVersion
---------------------- --------------

    -------------------------
AzureCatalog           XDHyp:\HostingUnits\AzureRes\machineprofile.
    folder\rg.resourcegroup\machineprofile.templatespec\1.0.
    templatespecversion                                        1

Set-ProvScheme -ProvisioningSchemeName AzureCatalog -MachineProfile
    XDHyp:\HostingUnits\AzureRes\machineprofile.folder\rg.resourcegroup\
    machineprofile.templatespec\2.0.templatespecversion

Get-ProvScheme | select ProvisioningSchemeName, MachineProfile,
    ProvisioningSchemeVersion

ProvisioningSchemeName MachineProfile

    ProvisioningSchemeVersion
---------------------- --------------

    -------------------------
AzureCatalog           XDHyp:\HostingUnits\AzureRes\machineprofile.
    folder\rg.resourcegroup\machineprofile.templatespec\2.0.
    templatespecversion                                        2

Set-ProvScheme -ProvisioningSchemeName MyScheme -Version 1

Get-ProvScheme | select ProvisioningSchemeName, MachineProfile,
    ProvisioningSchemeVersion

ProvisioningSchemeName MachineProfile

    ProvisioningSchemeVersion
---------------------- --------------

    -------------------------
AzureCatalog           XDHyp:\HostingUnits\AzureRes\machineprofile.
    folder\rg.resourcegroup\machineprofile.templatespec\1.0.
    templatespecversion                                        1
```


EXAMPLE 4

Updates a provisioning scheme called “MyScheme”to explicitly attach NICs to the network named “Network 2”for new VMs that are created from the provisioning scheme.

```
Set-ProvScheme -ProvisioningSchemeName MyScheme -NetworkMapping @{
 "0" = "XDHyp:\HostingUnits\XenHU\Network 2.network" }
```

EXAMPLE 5

Updates a provisioning scheme called “MyScheme”to use the machine profile from the aws ec2 instance - “machine-profile-vm (i-00c55f1107b662c04)”.

```
Set-ProvScheme -ProvisioningSchemeName MyScheme -MachineProfile "XDHyp
    :\HostingUnits\AwsHostingUnit\us-east-1a.availabilityzone\machine-
    profile-vm (i-00c55f1107b662c04).vm"
```

EXAMPLE 6

Updates a provisioning scheme called “MyScheme”to use the machine profile from the version “1”of the aws launch template with name “machine-profile-lt”and id “lt-06927522c36bg5698”.

```
Set-ProvScheme -ProvisioningSchemeName MyScheme -MachineProfile "XDHyp
    :\HostingUnits\AwsHostingUnit\machine-profile-lt (lt-06927522
    c36bg5698).launchtemplate\lt-06927522c36bg5698 (1).
    launchtemplateversion"
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to be updated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool to be used for the provisioning scheme, replacing the present one.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier of the identity pool to be used for the provisioning scheme, replacing the present one.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeUid

The identifier of the provisioning scheme to be updated.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Version

Specifies a prior version to change the provisioning scheme state to. Cannot be used in conjunction with other parameters that change the provisioning scheme configuration (e.g. -CustomProperties).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMCpuCount

The number of processors that will be used to create VMs from the provisioning scheme, replacing the present one.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMMemoryMB

The maximum amount of memory that will be used to created VMs from the provisioning scheme in MB, replacing the present one.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-CustomProperties

The properties of the provisioning scheme that are specific to the target hosting infrastructure. See about_ProvCustomProperties for more information. If a property name already exists its value is updated; otherwise it is added.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceOffering

The Service Offering to use when creating VMs in Cloud Hypervisors, replacing the present one.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineProfile

Currently only supported with Azure and AWS. Defines the inventory path to the source VM used by the provisioning scheme as a template. This profile identifies the properties for the VMs created from the scheme. The VM must be in the hosting unit that HostingUnitName or HostingUnitUid refers to. If any properties are present in the MachineProfile but not the CustomProperties, values from the template will be written back to the CustomProperties.

Valid paths are of the format: XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkMapping

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, the current NICs setting of the provisioning scheme is not updated; otherwise the NICs setting is updated, and new machines will be created with the number of NICs specified in the map, with each NIC attached to the specified network. Cannot set to an empty value.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SecurityGroup

The Security Groups to use when creating VMs in Cloud Hypervisors, replacing the present one.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheDiskSize

Specifies the size in Gigabytes of the disk to use as a Write Back Cache when UseWriteBackCache is set during the Provisioning Scheme creation. This parameter can only be set or modified if the Provisioning Scheme was previously created with UseWriteBackCache. This parameter only applies to newly created VMs and does not affect VMs which have already been created from the Provisioning Scheme.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheMemorySize

Specifies the size in Megabytes of the memory to use as a Write Back Cache when UseWriteBackCache is set during the Provisioning Scheme creation. This parameter can only be set or modified if the

Provisioning Scheme was previously created with UseWriteBackCache. This parameter only applies to newly created VMs and does not affect VMs which have already been created from the Provisioning Scheme. Setting this parameter to 0 disables the use of memory for Write Back Cache.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

This object provides details of the provisioning scheme and contains the following information:

CleanOnBoot <bool> Indicates whether the VMs created will be reset to a clean state on each start.

ControllerAddress <string[]> The DNS names of the controllers associated with this provisioning scheme.

CpuCount <int> The number of processors that will be used to create VMs.

DiskSize <int> The disk size (in GB) that will be used to create VMs.

HostingUnitName <string> The name of the hosting unit being used by this provisioning scheme.

HostingUnitUid <Guid> The unique identifier of the hosting unit being used by this provisioning scheme.

IdentityPoolName <string> The name of the identity pool being used by this provisioning scheme.

IdentityPoolUid <Guid> The unique identifier of the identity pool being used by this provisioning scheme.

MachineCount <int> The count of machines created with this provisioning scheme.

MachineProfile <string> The inventory path to the source VM used by the provisioning scheme as a template.

MasterImageVM <string> The inventory path to the VM snapshot copy used by this provisioning scheme.

MasterImageVMDate <DateTime> The date and time when the VM snapshot copy used by this provisioning scheme was made.

MemoryMB <int> The maximum amount of memory that will be used to created VMs in MB.

Metadata <Citrix.MachineCreation.Sdk.Metadata[]> The metadata associated with this provisioning scheme.

MetadataMap <IDictionary[string, string];> The metadata associated with this provisioning scheme arranged in key value pairs.

ProvisioningSchemeName <string> The name of this provisioning scheme.

ProvisioningSchemeUid <Guid> The unique identifier for this provisioning scheme.

ProvisioningSchemeVersion <int> The version of the provisioning scheme.

TaskId <Guid> The identifier of any current task that is running for the provisioning scheme.

VMMetadata <char[]> The metadata that will be used to created VMs in a plain text format.

PersonalVDiskDriveLetter <char> The drive letter for the personal vDisk.

PersonalVDiskDriveSize <int> The size of the personal vDisk in GB.

UsePersonalVDiskStorage <bool> Indicates whether this provisioning scheme uses personal vDisk storage.

NetworkMaps <Citrix.MachineCreation.Sdk.NetworkMap[]> The NIC/network mappings that will be used to create VMs.

Scope <Citrix.MachineCreation.Sdk.ScopeReference[]> The administration scopes associated with this provisioning scheme.

DedicatedTenancy <bool> Indicates whether dedicated tenancy is used when creating VMs in Cloud Hypervisors.

GpuTypeId <string> The id of the GPU (Graphics Processor Unit) Type used by this scheme. It is null if there is no GPU.

ResetAdministratorPasswords <bool> Indicates whether the passwords for administrator accounts are reset on created machines.

ServiceOffering <string> The service offering that the scheme uses when creating VMs in Cloud Hypervisors.

SecurityGroups <string[]> The security groups that will be applied to machines created in Cloud Hypervisors.

TenancyType <string> Tenancy type to be used when creating VMs in Cloud Hypervisors. (See New-ProvScheme.)

AzureAdJoinType <string> Deprecated.

CurrentMasterImageUid <Guid> The unique identifier of the current master image used by the provisioning scheme. (See Get-ProvSchemeMasterVMImageHistory.)

CustomProperties <string> Properties of the provisioning scheme which that are specific to the target hosting infrastructure. (See about_ProvCustomProperties)

IdentityType <string> Identity type used to join created machines to a directory service. (See New-ProvScheme.)

UseFullDiskCloneProvisioning <bool> Indicates whether VMs will be created using the dedicated full disk clone feature.

UseWriteBackCache <bool> Indicates whether this provisioning scheme will use the write-back cache feature.

WriteBackCacheDiskSize <int> The size of the write-back cache disk to be used in GB. Specify only when UseWriteBackCache is true.

WriteBackCacheMemorySize <int> The size of the write-back memory cache in MB. Specify only when UseWriteBackCache is true.

Warnings <Citrix.MachineCreation.Sdk.ProvSchemeWarning[]> Warning states that have occurred with this provisioning scheme.

WriteBackCacheDiskIndex <int> The disk index for the write-back cache disk. Specify only when UseWriteBackCache is true.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound The specified provisioning scheme could not be located.

DatabaseError An error occurred in the service while attempting a database operation.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

CommunicationError An error occurred while communicating with the service.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

HostingUnitNotFound The hosting unit referenced by the provisioning scheme could not be resolved.

NetworkPathResolutionFailed The specified network mapping contains invalid or empty network path value.

InvalidMachineProfileMetadata Unable to read the metadata from machine profile.

MachineProfileNotSupportedByHypervisor The hypervisor does not support any functional capability for MachineProfile.

MachineProfileUpdateNotSupported Adding a MachineProfile to the catalog is not supported.

IncorrectTenancyType MachineProfile TenancyType does not match with provisioning scheme.

ServiceOfferingPathResolutionFailed The Service Offering path could not be resolved. Please ensure that the path includes a drive specification and path to a location within a HostingUnit.

SecurityGroupPathResolutionFailed The Security Group path could not be resolved. Please ensure that the path includes a drive specification and path to a location within a HostingUnit.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

InvalidProvisioningSchemeVersionMaximum The provisioning scheme maximum is set to an invalid value. Value values are in the range (1, 32000).

ProvisioningSchemeVersionMaximumExceeded The change in provisioning scheme could not be performed due to the maximum version number being exceeded. Older versions were not able to be pruned.

Each time Set-ProvScheme is called, a snapshot of the current configuration is saved. To view the prior saved configurations, use Get-ProvVMConfiguration. To adjust the maximum number of prior configurations MCS saves, use Set-ProvServiceConfigurationData to change MaxProvSchemeVersions.

Running Set-ProvScheme to update provisioning scheme properties will only affect future provisioned machines. To have existing VMs in a catalog be brought up to date, use Set-ProvVMUpdateTimeWindow to set a time window for certain or all VMs created from the provisioning scheme associated with the catalog to be updated on next power on within the given time window.

Related Links

- about_ProvMachineCreationSnapin
- about_Prov_CustomProperties
- New-ProvScheme
- Remove-ProvScheme
- Get-ProvScheme
- Rename-ProvScheme
- Get-ProvSchemeVersion
- Set-ProvVMUpdateTimeWindow
- Clear-ProvVMUpdateTimeWindow
- Set-ProvServiceConfigurationData

Set-ProvSchemeImage

March 11, 2024

Update the master image associated with the provisioning scheme.

Syntax

```
Set-ProvSchemeImage
   [-ProvisioningSchemeName] <String>
   -PreparedImageDefinitionName <String>
   -PreparedImageVersionNumber <String>
   [-DoNotStoreOldImage]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeImage
   -ProvisioningSchemeUid <Guid>
   -PreparedImageVersionUid <Guid>
   [-DoNotStoreOldImage]
   [-RunAsynchronously]
   [-PurgeJobOnSuccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to update the hard disk image used to provision virtual machines. If the provisioning scheme is a 'CleanOnBoot'type, then the next time that virtual machines are started, their hard disks are updated to this new image. Regardless of the 'CleanOnBoot'type, all new virtual machines created after this command will use this new hard disk image.

A background task will be created to remove the old hard disk copies. You can locate and monitor this task using the Get-ProvTask cmdlet.

A snapshot or VM template is used rather than a VM, so that the content of the hard disk for the provisioning scheme can be easily determined.

As the snapshot or VM template are specified using a path into the Citrix Host Service PowerShell Provider, the Citrix Host Service PowerShell snap-in must also be loaded for this cmdlet to be used.

The previous hard disk image path is stored into the history (see Get-ProvSchemeMasterVMImageHistory). The data stored in the history allows for a rollback to be undertaken, to revert to the previous hard disk image if required.

Examples

EXAMPLE 1

Updates the image version for the provisioning Scheme "MyScheme"to use the image definition myImage version 2.

```
Set-ProvSchemeImage -ProvisioningSchemeName MyScheme -
    PreparedImageDefinitionName myImage -PreparedImageVersionNumber 2

TaskId                 : 248f102f-073f-45fe-aea9-1819a6d6dd1f
Active                 : False
Host                   : MyHost
DateStarted            : 17/05/2010 17:37:57
Type                   : SetProvisioningSchemeImage
Status                 :
CurrentOperation       :
TaskExpectedCompletion : 17/05/2010 17:38:43
LastUpdateTime         : 17/05/2010 17:38:43
ActiveElapsedTime      : 46
DateFinished           : 17/05/2010 17:38:43
TerminatingError       :
WorkflowStatus         : Completed
ProvisioningSchemeUid  : 7585f0de-192e-4847-a6d8-22713c3a2f42
ProvisioningSchemeName : MyScheme
ImageVersionNumber     : 2
ImageVersionUid        : 5991ba52-cad1-4e9c-b34f-1a6156e5f140
ImageDefinitionName    : myImage
ImageDefinitionUid     : 42bd09f5-3fac-44a6-8b1c-7c09f1079cc0
MasterImage            : /RhoneCC_baseXP.vm/base.snapshot
HostingUnitName        : HostUnit1
HostingUnitUid         : 01a4a008-8ce8-4165-ba9c-cdf15a6b0501
TaskState              : Finished
TaskStateInformation   : Completed
TaskProgress           : 100
Warnings               : {
  }

DiskSize               : 32
```


Parameters

-ProvisioningSchemeName

The provisioning scheme to which the hard disk image should be updated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-PreparedImageDefinitionName

The name for the image definition used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionNumber

The version number for the image version used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-ProvisioningSchemeUid

The provisioning uid identifier to which the hard disk image should be updated.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PreparedImageVersionUid

The identifier for the image definition used for the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

Indicates whether or not the cmdlet should return before it is complete. If specified, the command returns an identifier for the task that was created. You can monitor this task using the Get-ProvTask cmdlet.

Type:	SwitchParameter
Position:	Named

Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PurgeJobOnSuccess

Indicates that the task history will be removed from the database once the task has finished. This cannot be specified for tasks that are run asynchronously.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DoNotStoreOldImage

Indicates that the current image should not be added to the image history list for the provisioning scheme. This is useful when rolling back to a previous image.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Guid

When "RunAsynchronously"is specified, this identifier is returned to provide the task identifier.

System.Management.Automation.PSCustomObject

This object provides details of the task run and contains the following information:

TaskId <Guid>

The identifier for the task that was performed.

Active <Boolean>

Indicates whether the task is still processing or is complete.

Host <string>

The name of the host on which the task is running or was run.

DateStarted <DateTime>

The date and time that the task was initiated.

Type <Citrix.XDInterServiceTypes.JobType>

The type of the task. For set provisioning scheme image tasks this is always "SetProvisioningSchemeImage".

Metadata <Citrix.MachineCreation.Sdk.Metadata[]>

The list of metadata stored against the task. For new tasks this will be empty until metadata is added.

WorkflowStatus <System.Workflow.Runtime.WorkflowStatus>

Indicates the status of the workflow that is being used to process the task.

ProvisioningSchemeName <string>

The name of the provisioning scheme that the task was for.

ProvisioningSchemeUid <Guid>

The unique identifier of the provisioning scheme that the task was for.

MasterImage <string>

The path (in the hosting unit provider) of the virtual machine snapshot or VM template that was used as the master image for the task.

IdentityPoolName <string>

The name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme will use.

IdentityPoolUid <guid>

The unique identifier name of the identity pool (from the ADIdentity PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitName <string>

The name of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

HostingUnitUid

The unique identifier of the hosting unit (from the Hosting Unit PowerShell snap-in) that the new provisioning scheme will use.

MasterImageNote

The note of the provisioning scheme image.

TaskState <Citrix.DesktopUpdateManager.SDK.NewProvisioningSchemeState>

The state of the task. This can be any of the following:

Processing

Indicates that the task has just begun and has not done anything yet.

LocatingResources,

Indicates that the workflow is locating resources from other services.

HostingUnitNotFound

Indicates that the task failed because the required hosting unit could not be located.

VirtualMachineSnapshotNotFound

Indicates that the task failed because the required snapshot or VM template could not be located.

ConsolidatingMasterImage

Indicates that the task is consolidating the master image.

ReplicatingConsolidatedImageToAllStorage

Indicates that the task is replicating the consolidated master image.

StoringProvisioningScheme

Indicates that the task is storing the provisioning scheme data to the database.

Finished

Indicates that the task has completed with no errors.

ProvisioningSchemeAlreadyExists

Indicates that the task failed because a provisioning scheme with the same name already exists.

IdentityPoolNotFound

Indicates that the task failed because the identity pool specified could not be found.

MasterVMImageIsNotPartOfProvisioningSchemeHostingUnit,

Indicates that the hosting unit that the master image originated from is not the same hosting unit that the provisioning scheme is defined to use.

MasterVmImageIsNotASnapshot

Indicates that the task failed because the master VM image path does not refer to a snapshot or a VM template item.

ProvisioningSchemeNotFound

Could not find a provisioning scheme with the specified name.

TaskAlreadyRunningForProvisioningScheme

A task for a provisioning scheme with the same name is already running.

InvalidMasterVMConfiguration

The VM snapshot or VM template specified as the master had an invalid configuration.

InvalidMasterVMState

The VM snapshot or VM template specified as the master is currently in an invalid state.

InsufficientResources

Indicates that the task failed because the hypervisor did not have enough resources to complete the task.

StorageNotFound

Indicates that no associated storage was found in the hosting unit.

Canceled

Indicates that the task was stopped by user intervention (using Stop-ProvTask).

TaskStateInformation <string>

Holds string data providing more details about the current task state.

TaskProgress

The progress of the task 0-100%.

Notes

Only one long-running task per provisioning scheme can be processed at any one time.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

JobCreationFailed

The requested task could not be started.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

WorkflowHostUnavailable

The task could not be started because the database connection is inactive.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

ExceptionThrown

An unexpected error occurred. To locate more details, see the Windows event logs on the controller being used or examine the XenDesktop logs.

The cmdlet is associated with a task of type SetProvisioningSchemeImage, and while active will move through the following operations (CurrentOperation field)

ValidatingInputs

ReplicatingMasterImage

CommittingScheme

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeMasterVMImageHistory
- Get-ProvScheme

Set-ProvSchemeMetadata

March 11, 2024

Adds or updates metadata on the given provisioning scheme.

Syntax

```
Set-ProvSchemeMetadata
   [-ProvisioningSchemeUid] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeMetadata
   [-ProvisioningSchemeUid] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeMetadata
   [-ProvisioningSchemeName] <String>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeMetadata
   [-ProvisioningSchemeName] <String>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeMetadata
```



```
    [-InputObject] <ProvisioningScheme[]>
    -Name <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvSchemeMetadata
    [-InputObject] <ProvisioningScheme[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given provisioning scheme objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the provisioning scheme with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ProvSchemeMetadata -ProvisioningSchemeUid 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                     Value
---                                     -----
property                                value
```

Parameters

-ProvisioningSchemeUid

Id of the provisioning scheme

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeName

Name of the provisioning scheme

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-InputObject

ProvisioningScheme objects to which the metadata is to be added or updated.

Type:	ProvisioningScheme[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added or updated. The property must be unique for the provisioning scheme specified. If the name already exists, its value is updated. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe a ProvisioningScheme object or any object containing a parameter called 'ProvisioningSchemeName'or 'ProvisioningSchemeUid'to Set-ProvSchemeMetadata.

PSObject

A metadata map object can be piped to the Set-ProvSchemeMetadata command.

Outputs

System.Collections.Generic.Dictionary[string, string]

Set-ProvSchemeMetadata returns a dictionary containing the new (name, value)-pairs for the metadata being set.

Key <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvSchemeMetadata
- Remove-ProvSchemeMetadata

Set-ProvSchemeWarning

March 11, 2024

Sets the WarningStateType of a list of provisioning schemes warnings.

Syntax

```
Set-ProvSchemeWarning
   [-ProvisioningSchemeName <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeWarning
   [-ProvisioningSchemeUid <Guid>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvSchemeWarning
   [-WarningId <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ProvSchemeWarning
    [-All]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to set the WarningStateType from New to Acknowledged.

Examples

EXAMPLE 1

Sets the warning with a WarningId of 42 to Acknowledged.

```
Set-ProvSchemeWarning -WarningId 42
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme for which all associated warnings will be set to Acknowledged.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme for which all associated warnings will be set to Acknowledged.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WarningId

The id of the warning to set to Acknowledged.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-All

All warnings for all provisioning schemes will be set to Acknowledged.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

PartialData Only a subset of the available data was returned

CouldNotQueryDatabase The query to get the database was not defined.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

CommunicationError An error occurred while communicating with the service.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

InvalidFilter A filtering expression was supplied that could not be interpreted for this cmdlet.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvSchemeWarning
- Remove-ProvSchemeWarning

Set-ProvServiceConfigurationData

March 11, 2024

Sets the value for the given key in the service configuration data.

Syntax


```
Set-ProvServiceConfigurationData
    [-Name] <String>
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to store configuration data for the Machine Creation Service.

Examples

EXAMPLE 1

Sets MaxProvSchemeVersions to 15 in the service configuration.

```
Set-ProvServiceConfigurationData -Name "MaxProvSchemeVersions" -Value
    15

Name                                Value
--------                            -----
MaxProvSchemeVersions                  15
```

Parameters

-Name

Specifies the key name of the configuration data to be added. The name must be unique. If the name already exists, its value is updated.

The name cannot contain any of the following characters \/;:#.*?=<>|[]()’”

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the name.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.ServiceConfigurationData

Set-ProvServiceConfigurationData returns an object containing the new definition of the configuration.

Name <string>

Specifies the name for the item of data.

Value <string>

Specifies the value of the data.

Notes

In the case of failure the following errors can be produced.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

An error occurred while communicating with the service.

ExceptionThrown

An unexpected error occurred. For more details see the Windows event logs on the controller being used, or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvServiceConfigurationData
- Get-ProvServiceConfigurationData

Set-ProvServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-ProvServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ProvServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-ProvServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-ProvServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Remove-ProvServiceMetadata

Set-ProvTaskMetadata

March 11, 2024

Adds or updates metadata on the given Task.

Syntax

```
Set-ProvTaskMetadata
   [-TaskId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvTaskMetadata
   [-TaskId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvTaskMetadata
   [-InputObject] <Task[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvTaskMetadata
   [-InputObject] <Task[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Use this cmdlet to store additional custom data against given Task object.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Task with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-ProvTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name
    property -Value value

Key                                     Value
```



```
4 ---                                          -----
5 property                                     value
```

Parameters

-TaskId

Id of the Task.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	Task[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[string, string];”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Set-ProvTaskMetadata command.

PSObject

A metadata map object can be piped to the Set-ProvTaskMetadata command.

Outputs

System.Collections.Generic.Dictionary[string, string]

Set-ProvTaskMetadata returns a dictionary containing the new (name, value)-pairs for the metadata being set.

Key <string>

Name of the property.

Value <string>

Value for the property.

Notes

If the command fails, the following errors can result.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Add-ProvTaskMetadata
- Remove-ProvTaskMetadata
- Get-ProvTask
- Stop-ProvTask
- Remove-ProvTask
- Switch-ProvTask

Set-ProvVM

March 11, 2024

Changes the configuration for a specific provisioned virtual machine.

Syntax

```
Set-ProvVM
   -ProvisioningSchemeName <String>
   -VMName <String>
   [-CpuCount <Int32>]
   [-MemoryInMB <Int32>]
   [-CustomProperties <String>]
   [-ServiceOffering <String>]
   [-MachineProfile <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeName <String>
   -VMId <String>
   [-CpuCount <Int32>]
   [-MemoryInMB <Int32>]
   [-CustomProperties <String>]
   [-ServiceOffering <String>]
   [-MachineProfile <String>]
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeName <String>
   -VMName <String>
   [-RevertToProvSchemeConfiguration]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeName <String>
   -VMId <String>
   [-RevertToProvSchemeConfiguration]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeUid <Guid>
   -VMName <String>
   [-CpuCount <Int32>]
   [-MemoryInMB <Int32>]
   [-CustomProperties <String>]
   [-ServiceOffering <String>]
   [-MachineProfile <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeUid <Guid>
   -VMId <String>
   [-CpuCount <Int32>]
   [-MemoryInMB <Int32>]
   [-CustomProperties <String>]
   [-ServiceOffering <String>]
   [-MachineProfile <String>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVM
   -ProvisioningSchemeUid <Guid>
   -VMName <String>
   [-RevertToProvSchemeConfiguration]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-ProvVM
   -ProvisioningSchemeUid <Guid>
   -VMId <String>
   [-RevertToProvSchemeConfiguration]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability to update the configuration of an existing persistent provisioned virtual machine.

These properties are specific customizations unique to the virtual machine and do not apply to any other VMs in the catalog. The final properties the machine will use are a combination of those currently set on the provisioning scheme combines with the settings specified here. Any properties set at the ProvVM-level with this command will override any settings at the ProvScheme-level.

To view the current specific configuration for a machine, use Get-ProvVMConfiguration

To view the final set of properties a machine will have applied (taking into account ProvScheme-level settings), use Get-ProvVMConfigurationResultantSet

Any configuration changes made will not apply right away to the machine. To apply the updates, set an update time window for the machine with Set-ProvVMUpdateTimeWindow and start the machine within the window. Currently, this feature is applicable only to the AWS and Azure environments.

Examples

EXAMPLE 1

Sets the configuration for machine1 to use 2 CPUs. Sets an update time window starting now and reboots the machine to apply the update.

```
Get-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 | select
   VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1


Get-ProvVMConfiguration

Set-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 -CpuCount
   2

Get-ProvVMConfiguration
```



```
CpuCount                : 2
CustomProperties        :
MachineProfile          :
MemoryInMB              :
ProvisioningSchemeName  : MyScheme
ProvisioningSchemeUid   : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering         :
VMId                    : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata              :
VMName                  : machine1
Version                 : 1

Get-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName MyScheme -StartsNow
    -DurationInMinutes -1

New-BrokerHostingPowerAction -Action "Restart" -MachineName machine1

Get-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1                        1
```

EXAMPLE 2

Clears existing configuration for machine1. Sets an update time window starting now and reboots the machine to clear the configuration.

```
Get-ProvVMConfiguration
CpuCount                : 8
CustomProperties        :
MachineProfile          :
MemoryInMB              : 4096
ProvisioningSchemeName  : MyScheme
ProvisioningSchemeUid   : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering         :
VMId                    : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata              :
VMName                  : machine1
Version                 : 7

Set-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 -
    RevertToProvSchemeConfiguration

Get-ProvVMConfiguration
```



```
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : MyScheme
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        :
VMId                   : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata             :
VMName                 : machine1
Version                : 8

Get-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1                       7

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName MyScheme -StartsNow
    -DurationInMinutes -1

New-BrokerHostingPowerAction -Action "Restart" -MachineName machine1

Get-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1                       8
```

EXAMPLE 3

Configures machine1 to have a custom MachineProfile. ServiceOffering and CustomProperties were read from MachineProfile and set on the configuration

```
Get-ProvVMConfiguration
CpuCount               :
CustomProperties       : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                           <Property xsi:type="StringProperty" Name="
                              WBCDiskStorageType" Value="Premium_LRS" />
                           </CustomProperties>
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : MyScheme
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        : serviceoffering.folder\Standard_D2s_v5.
    serviceoffering
VMId                   : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata             :
VMName                 : machine1
```



```
Version                : 1

Set-ProvVM -ProvisioningSchemeName MyScheme -VMName machine1 -
    MachineProfile "XDHyp:\HostingUnits\AzureRes\machineprofile.folder\
    TestRG.resourcegroup\machineProfile.vm"

Get-ProvVMConfiguration
CpuCount               :
CustomProperties       : <CustomProperties xmlns="http://schemas.citrix
    .com/2014/xd/machinecreation" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance">
                           <Property xsi:type="StringProperty" Name="
                              StorageType" Value="Premium_LRS" />
                           <Property xsi:type="StringProperty" Name="
                              WBCDiskStorageType" Value="Premium_LRS" />
                           </CustomProperties>
MachineProfile         : machineprofile.folder\TestRG.resourcegroup\
    machineProfile.vm
MemoryInMB             :
ProvisioningSchemeName : MyScheme
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        : serviceoffering.folder\Standard_D2s_v3.
    serviceoffering
VMId                   : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata             :
VMName                 : machine1
Version                : 2
```

EXAMPLE 4

In this AWS example Set-ProvVM sets the configuration for machine1 to use T3 Large ServiceOffering. Sets an update time window starting now and reboots the machine to apply the update.

```
Get-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1

Get-ProvVMConfiguration AWSScheme -VMName machine1

Set-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 -
    ServiceOffering "XDHyp:\HostingUnits\AWSHostingUnit\T3 Large
    Instance.serviceoffering"

Get-ProvVMConfiguration AWSScheme -VMName machine1
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : AWSScheme
```



```
ProvisioningSchemeUid  : a13a9d94-a2a1-46ee-8363-1d225e9ef468
ServiceOffering        : T3 Large Instance.serviceoffering
VMId                   : i-0bdd24b3309920f3a
VMMetadata             :
VMName                 : machine1
Version                : 1

Get-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName     ProvVMConfigurationVersion
------     -------
machine1

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName AWSScheme -VMName
    machine1 -StartsNow -DurationInMinutes -1

New-BrokerHostingPowerAction -Action "Restart" -MachineName machine1

Get-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName     ProvVMConfigurationVersion
------     -------
machine1   1
```

EXAMPLE 5

In this AWS example -RevertToProvSchemeConfiguration clears existing configuration for machine1. Sets an update time window starting now and reboots the machine to clear the configuration.

```
Get-ProvVMConfiguration AWSScheme -VMName machine1
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeUid  : a13a9d94-a2a1-46ee-8363-1d225e9ef468
ServiceOffering        : T3 Large Instance.serviceoffering
VMId                   : i-0bdd24b3309920f3a
VMMetadata             :
VMName                 : machine1
Version                : 1

Set-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 -
    RevertToProvSchemeConfiguration


Get-ProvVMConfiguration AWSScheme -VMName machine1
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : AWSScheme
```



```
ProvisioningSchemeUid  : a13a9d94-a2a1-46ee-8363-1d225e9ef468
ServiceOffering        :
VMId                   : i-0bdd24b3309920f3a
VMMetadata             :
VMName                 : machine1
Version                : 2

Get-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName     ProvVMConfigurationVersion
------     -------
machine1   1

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName AWSScheme -StartsNow
     -DurationInMinutes -1

New-BrokerHostingPowerAction -Action "Restart" -MachineName machine1

Get-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 | select
    VMName, ProvVMConfigurationVersion
VMName      ProvVMConfigurationVersion
------      --------------------
machine1                       2
```

EXAMPLE 6

In this AWS example Set-ProvVM configures machine1 to have a custom MachineProfile. ServiceOffering was read from MachineProfile and set on the configuration

```
Get-ProvVMConfiguration AWSScheme -VMName machine1
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : AWSScheme
ProvisioningSchemeUid  : a13a9d94-a2a1-46ee-8363-1d225e9ef468
ServiceOffering        :
VMId                   : i-0bdd24b3309920f3a
VMMetadata             :
VMName                 : machine1
Version                : 2

Set-ProvVM -ProvisioningSchemeName AWSScheme -VMName machine1 -
    MachineProfile "XDHyp:\HostingUnits\AWSHostingUnit\LaunchTemplate1.
    launchtemplate\lt-version (4).launchtemplateversion"

Get-ProvVMConfiguration AWSScheme -VMName machine1
CpuCount               :
CustomProperties       :
MachineProfile         : LaunchTemplate1.launchtemplate\lt-version (4).
    launchtemplateversion
```



```
MemoryInMB             :
ProvisioningSchemeName : AWSScheme
ProvisioningSchemeUid  : a13a9d94-a2a1-46ee-8363-1d225e9ef468
ServiceOffering        : T3 Medium Instance.serviceoffering
VMId                   : i-0bdd24b3309920f3a
VMMetadata             : {
 A, A, E, A... }

VMName                 : machine1
Version                : 3
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme the VM is a part of

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMName

The VM name

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMId

The VM ID

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RevertToProvSchemeConfiguration

If supplied, all existing configuration is cleared for the given machine. It is mutally exclusive with parameters that apply configuration settings

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeUid

The identifier of the provisioning scheme the VM is a part of

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-CpuCount

The number of processors. Not supported for cloud hypervisors

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MemoryInMB

The maximum amount of memory in megabytes. Not supported for cloud hypervisors

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CustomProperties

The properties of the provisioned virtual machine that are specific to the target hosting infrastructure. See about_ProvCustomProperties for more information. If a property name already exists its value is updated; otherwise it is added. These properties are merged with prior custom properties set.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-ServiceOffering

The cloud ServiceOffering to use. Not supported for on-prem hypervisors

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineProfile

A path to the template used to obtain hypervisor-specific settings to be applied to the VM. Some settings have a corresponding CustomProperty. If any properties are present in the MachineProfile but not the CustomProperties, values from the template will be written to the CustomProperties.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- about_Prov_CustomProperties
- Get-ProvVM
- Get-ProvVMConfiguration
- Get-ProvVMConfigurationResultantSet
- Set-ProvVMUpdateTimeWindow

Set-ProvVMUpdateTimeWindow

March 11, 2024

Sets a time window on provisioned virtual machines during which they will undergo a property update on boot.

Syntax

```
Set-ProvVMUpdateTimeWindow
   [-StartTimeInUTC <String>]
   [-StartsNow]
   [-DurationInMinutes <Int32>]
   -ProvisioningSchemeName <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-ProvVMUpdateTimeWindow
   [-StartTimeInUTC <String>]
   [-StartsNow]
   [-DurationInMinutes <Int32>]
   -ProvisioningSchemeName <String>
   -VMName <String[]>
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvVMUpdateTimeWindow
    [-StartTimeInUTC <String>]
    [-StartsNow]
    [-DurationInMinutes <Int32>]
    -ProvisioningSchemeUid <Guid>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-ProvVMUpdateTimeWindow
    [-StartTimeInUTC <String>]
    [-StartsNow]
    [-DurationInMinutes <Int32>]
    -ProvisioningSchemeUid <Guid>
    -VMName <String[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to synchronize the properties of existing virtual machines with any changes occurred on the provisioning scheme with Set-ProvScheme and any configuration applied with Set-ProvVM.

This cmdlet defines a time window where a property update will take place against either specific machines in a catalog or an entire catalog. The update will take place when the machine is next booted within the time window. Note that running this command does not make the machine automatically reboot or turn on sometime within the time window to apply the update. The power action to initiate the update must either be performed as a part of an autoscale schedule or manually through Studio or the New-BrokerHostingPowerAction cmdlet within the time window. If the time window passes before the machine is booted successfully, the update will not apply and will have to be set again with this cmdlet.

On each VM an update window is set, ProvisioningSchemeUpdateRequested defines the start of the time window and ProvisioningSchemeUpdateUntil defines the end of the time window.

After the update, ProvisioningSchemeVersion on the VM will be updated to match the current version of the ProvScheme, and the ProvVMConfigurationVersion will match the latest configuration on the VM.

Examples

EXAMPLE 1

Change the MachineProfile for the azure-catalog provisioning scheme with Set-ProvScheme, this increments the provisioning scheme version to 2. Set all 4 VMs in azure-catalog to have a property update in the time window starting 3/12/2022 at 3am UTC and lasting for 60 minutes. Times for ProvisioningSchemeUpdateRequested and ProvisioningSchemeUpdateUntil are displayed in local time (EST in this case). If a catalog machine is booted in that time window, it will update to the latest provisioning scheme settings. The first restart of machine azu01 in this example performs no action since it's before the time window starts, while the second restart is within the window, so the machine is updated and the time window is cleared.

```
Set-ProvScheme -ProvisioningSchemeName azure-catalog -MachineProfile "
    XDHyp:\HostingUnits\azure-unit\machineprofile.folder\new-template.vm
    "

Get-ProvScheme | select ProvisioningSchemeName,
    ProvisioningSchemeVersion
ProvisioningSchemeName   ProvisioningSchemeVersion
----------------------   --------------------------
azure-catalog                                    2

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
    ---------------------------------  -----------------------------
azu01                                1
azu02                                1
azu03                                1
azu04                                1

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog -
    StartTimeInUTC "3/12/2022 3am" -DurationInMinutes 60

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
    ---------------------------------  -----------------------------
azu01                                1              3/11/2022  11:00:00
    PM         3/12/2022  12:00:00 AM
azu02                                1              3/11/2022  11:00:00
    PM         3/12/2022  12:00:00 AM
azu03                                1              3/11/2022  11:00:00
    PM         3/12/2022  12:00:00 AM
azu04                                1              3/11/2022  11:00:00
```



```
    PM          3/12/2022  12:00:00 AM
25
26  Get-Date
27
28  Thursday, March 10, 2022 1:53:41 PM
29
30  New-BrokerHostingPowerAction -Action "Restart" -MachineName azu01
31
32  Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
33  VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
34  ------       -------------------------
    ---------------------------------  ----------------------------
35  azu01                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
36  azu02                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
37  azu03                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
38  azu04                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
39
40  Get-Date
41
42  Friday, March 11, 2022 11:53:41 PM
43
44  New-BrokerHostingPowerAction -Action "Restart" -MachineName azu01
45
46  Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
47  VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
48  ------       -------------------------
    ---------------------------------  ----------------------------
49  azu01                             2
50  azu02                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
51  azu03                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
52  azu04                             1               3/11/2022  11:00:00
    PM          3/12/2022  12:00:00 AM
```

EXAMPLE 2

Set azu01 and azu02 in azure-catalog to have a property update anytime from 11pm EST (-4) until 2 days later. The other machines remain having no property update time window set. Times for ProvisioningSchemeUpdateRequested and ProvisioningSchemeUpdateUntil are displayed in local time (EST in this case). The update will be run next time each machine boots in that window.


```
Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
    ---------------------------------  -----------------------------
azu01                                1
azu02                                1
azu03                                1
azu04                                1

$duration = New-TimeSpan -Days 2

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog -
    VMName azu01, azu02 -StartTimeInUTC "3/11/2022 11pm -4" -
    DurationInMinutes $duration.TotalMinutes

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
    ---------------------------------  -----------------------------
azu01                                1               3/11/2022  11:00:00
    PM         3/13/2022  11:00:00 PM
azu02                                1               3/11/2022  11:00:00
    PM         3/13/2022  11:00:00 PM
azu03                                1
azu04                                1
```

EXAMPLE 3

Set azu01 and azu02 in azure-catalog to have a property update anytime after 7 days from now. The other machines remain having no property update time window set. The update will be run next time each machine boots after the start time.

```
Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
------       -------------------------
    ---------------------------------  -----------------------------
azu01                                1
azu02                                1
azu03                                1
azu04                                1

$startTime = (Get-Date).AddDays(7).ToUniversalTime()

Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog -
    VMName azu01, azu02 -StartTimeInUTC $startTime -DurationInMinutes -1
```



```
12
13 Get-ProvVM | select VMName, ProvisioningSchemeVersion,
       ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
14 VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
15 ------       -------------------------
       ---------------------------------  ----------------------------
16 azu01                                1              3/18/2022  11:32:51
       PM
17 azu02                                1              3/18/2022  11:32:51
       PM
18 azu03                                1
19 azu04                                1
```

EXAMPLE 4

Set azu01 and azu02 in azure-catalog to have a property update when next booted. The other machines remain having no property update time window set.

```
1  Get-ProvVM | select VMName, ProvisioningSchemeVersion,
       ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
2  VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
3  ------       -------------------------
       ---------------------------------  ----------------------------
4  azu01                                1
5  azu02                                1
6  azu03                                1
7  azu04                                1
8
9  Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog -
       VMName azu01, azu02 -StartsNow  -DurationInMinutes -1
10
11 Get-ProvVM | select VMName, ProvisioningSchemeVersion,
       ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil
12 VMName       ProvisioningSchemeVersion
       ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
13 ------       -------------------------
       ---------------------------------  ----------------------------
14 azu01                                1              3/11/2022  11:32:51
       PM
15 azu02                                1              3/11/2022  11:32:51
       PM
16 azu03                                1
17 azu04                                1
```


EXAMPLE 5

Change the MachineProfile for the azure-catalog provisioning scheme with Set-ProvScheme, this increments the provisioning scheme version to 2. Configure a custom service offering for azu01. Set all 4 VMs in azure-catalog to have a property update in the time window starting now and lasting for 60 minutes. Times for ProvisioningSchemeUpdateRequested and ProvisioningSchemeUpdateUntil are displayed in local time (EST in this case). Rebooting azu01 applies both the pending configuration and provisioning scheme updates. Rebooting azu02 applies the pending provisioning scheme update

```
Set-ProvScheme -ProvisioningSchemeName azure-catalog -MachineProfile "
    XDHyp:\HostingUnits\azure-unit\machineprofile.folder\new-template.vm
    "

Get-ProvScheme | select ProvisioningSchemeName,
    ProvisioningSchemeVersion
ProvisioningSchemeName   ProvisioningSchemeVersion
----------------------  --------------------------
azure-catalog                                    2

Set-ProvVM -ProvisioningSchemeName azure-catalog -VMName azu01 -
    ServiceOffering "XDHyp:\HostingUnits\azure-unit\"

Get-ProvVMConfiguration
CpuCount               :
CustomProperties       :
MachineProfile         :
MemoryInMB             :
ProvisioningSchemeName : azure-catalog
ProvisioningSchemeUid  : 378cece5-a824-41f7-9e92-74be76672be6
ServiceOffering        : serviceoffering.folder\Standard_D2s_v3.
    serviceoffering
VMId                   : 0707da6d-2f0f-a8c7-ce92-3d64f824ac60
VMMetadata             :
VMName                 : azu01
Version                : 1


Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil,
    ProvVMConfigurationVersion
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
    ProvVMConfigurationVersion
------       -------------------------
    ---------------------------------  -----------------------------
    --------------------
azu01                                1
azu02                                1
azu03                                1
azu04                                1

```



```
Set-ProvVMUpdateTimeWindow -ProvisioningSchemeName azure-catalog -
    StartsNow -DurationInMinutes 60

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil,
    ProvVMConfigurationVersion
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
    ProvVMConfigurationVersion
------       -------------------------
    ---------------------------------  ----------------------------
    --------------------
azu01                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM
azu02                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM
azu03                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM
azu04                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM

New-BrokerHostingPowerAction -Action "Restart" -MachineName azu01

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil,
    ProvVMConfigurationVersion
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
    ProvVMConfigurationVersion
------       -------------------------
    ---------------------------------  ----------------------------
    --------------------
azu01                                2

    1
azu02                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM
azu03                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM
azu04                                1               3/11/2022  10:37:14
    PM         3/11/2022 11:37:14 PM

New-BrokerHostingPowerAction -Action "Restart" -MachineName azu02

Get-ProvVM | select VMName, ProvisioningSchemeVersion,
    ProvisioningSchemeUpdateRequested, ProvisioningSchemeUpdateUntil,
    ProvVMConfigurationVersion
VMName       ProvisioningSchemeVersion
    ProvisioningSchemeUpdateRequested  ProvisioningSchemeUpdateUntil
    ProvVMConfigurationVersion
------       -------------------------
    ---------------------------------  ----------------------------
    --------------------
```



```
azu01                                2


      1
azu02                                2
azu03                                1               3/11/2022  10:37:14
   PM         3/11/2022 11:37:14 PM
azu04                                1               3/11/2022  10:37:14
   PM         3/11/2022 11:37:14 PM
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-VMName

List of VM names set a time window for. If not specified, all VMs in the machine catalog associated with the specified provisioning scheme will have the time window applied

Type:	String[]

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StartTimeInUTC

The start time in UTC format of the time window. Mutually exclusive with StartsNow switch. Must be no more than 5 seconds in the past. The given string will be converted into a DateTime using parsing rules defined by the DateTime.Parse method, which allows for a timezone offset to specify time in any timezone. If no timezone offset is specified, UTC will be assumed. See the example usages of Set-ProvVMUpdateTimeWindow or this link for more details https://docs.microsoft.com/en-us/dotnet/api/system.datetime.parse?view=net-6.0#the-string-to-parse

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StartsNow

Whether this time window should start right away at the current time. Mutually exclusive with specific StartTimeInUTC. Implied to be set if StartTimeInUTC is not specified

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DurationInMinutes

The duration in minutes of the time window. If less than 0, indicates the time window has no upper bound

Type:	Int32
Position:	Named
Default value:	120 minutes
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId'and 'ProvisioningSchemeName'to Set-ProvVMUpdateTimeWindow.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Set-ProvScheme
- Set-ProvVM
- Clear-ProvVMUpdateTimeWindow

Stop-ProvTask

March 11, 2024

Stops currently running Machine Creation Service tasks.

Syntax

```
Stop-ProvTask
    [-TaskId] <Guid>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables tasks currently running in the Machine Creation Service to be stopped. Once stopped, tasks cannot be resumed.

Examples

EXAMPLE 1

Terminate all tasks currently executing on the Machine Creation Service.


```
Get-ProvTask -active $true | Stop-ProvTask
Success
```

EXAMPLE 2

Terminate the named task executing on the Machine Creation Service.

```
Stop-ProvTask -TaskId bd52e688-e40d-4790-83de-9f7633481454
Success
```

Parameters

-TaskId

Specifies the identifier for the task to be stopped.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Stop-ProvTask command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can result.

Error Codes

InvalidWorkflow

The specified task could not be found.

InvalidWorkflowHost

The specified task is executing on a different server.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

DatabaseError

There was a problem communicating with the database.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvTask
- Remove-ProvTask
- Switch-ProvTask

Switch-ProvTask

March 11, 2024

Moves all Machine Creation Service tasks from the current execution host to another.

Syntax

```
Switch-ProvTask
       [-Host2] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Enables tasks running within the Machine Creation Service to be moved from one execution host to a different host.

Tasks can only execute on a single host. If the host is removed from a deployment, the tasks cannot continue to execute. These 'orphaned'tasks can be moved to a different host within the deployment so that they can continue to execute. All tasks must be moved. There is no mechanism to move individual tasks.

Run the Switch-ProvTask command against the host to which the tasks are to be moved. If you are not running the command directly on the host, use the AdminAddress parameter to specify the host to which tasks will be moved.

Examples

EXAMPLE 1

Transfer the execution of tasks that had been executing on the Machine Creation Service instance on host CRASHED to the local instance.

```
Switch-ProvTask -Host CRASHED
Success
```

Parameters

-Host2

Specifies the host from which the tasks should be removed.

Type:	String
Aliases:	Host
Position:	2

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can result.

Error Codes

PartialData

Only a subset of the requested data was returned.

NoOp

The operation was successful but had no effect.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvTask
- Stop-ProvTask
- Remove-ProvTask

Test-ProvDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix MachineCreation Service.

Syntax

```
Test-ProvDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix MachineCreation Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a MachineCreation SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-ProvDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix MachineCreation Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-ProvDBConnection cmdlet returns an object describing the status of the selected MachineCreation Service instance that would result if the connection string were used with the Set-ProvDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-ProvDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the MachineCreation Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the MachineCreation Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The MachineCreation Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The MachineCreation Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the MachineCreation Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the MachineCreation Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvServiceStatus
- Get-ProvDBConnection
- Set-ProvDBConnection

Test-ProvInventoryItem

March 11, 2024

Check if an inventory item can be used as an input for machine profile.

Syntax

```
Test-ProvInventoryItem
    -HostingUnitName <String>
    -InventoryPath <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvInventoryItem
    -HostingUnitUid <Guid>
    -InventoryPath <String>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to validate if an inventory item can be used as an input for the MachineProfile parameter while creating or updating a provisioning scheme.

Examples

EXAMPLE 1

Validates if the Azure template spec inventory item located at XDHyp:\HostingUnits\AzureRes\machineprofile.folde can be used as input for MachineProfile to either New-ProvScheme or Set-ProvScheme. Since the cmdlet returns no response, the template is a valid MachineProfile.


```
Test-ProvInventoryItem -HostingUnitName AzureRes -InventoryPath
    machineprofile.folder\rg.resourcegroup\machineprofile.templatespec
    \1.0.templatespecversion
```

EXAMPLE 2

Validates if version 1.0 of the Azure template spec machineprofile found under the hosting unit with identifier 8101028f-2570-4e17-9129-8e0cfc1e2558 can be used as input for MachineProfile to either New-ProvScheme or Set-ProvScheme. The cmdlet returns some validation errors that must be fixed before the template can be used.

```
Test-ProvInventoryItem -HostingUnitUid 8101028f-2570-4e17-9129-8
    e0cfc1e2558 -InventoryPath machineprofile.folder\rg.resourcegroup\
    machineprofile.templatespec\1.0.templatespecversion

ErrorMessage
---------------
Error: Provided disk encryption set ID in MachineProfile (input from
    the Azure template spec) does not exist
Error: Invalid storage account type in MachineProfile (input from the
    Azure template spec)
```

Parameters

-HostingUnitName

The name of the hosting unit used for the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-HostingUnitUid

The identifier for the hosting unit used for the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InventoryPath

The partial inventory path to be validated as a machine profile. It should only contain the relative path from the hosting unit given in either the HostingUnitName or HostingUnitUid parameters. See the examples for how specify this partial path.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.MachineCreation.Sdk.InventoryItemValidationResult

This object provides a validation error message from the inventory item and contains the following information:

ErrorMessage <string>

String describing a validation error from the inventory item.

Notes

In the case of failure, the following errors can result.

Error Codes

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

Test-ProvSchemeNameAvailable

March 11, 2024

Check if the proposed provisioning scheme name is unused.

Syntax

```
Test-ProvSchemeNameAvailable
    -ProvisioningSchemeName <String[]>
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Provides the ability to validate if the proposed name for a provisioning scheme is unused. This check is not limited to scopes of existing provisioning schemes, therefore, the names of inaccessible schemes are also checked.

Examples

EXAMPLE 1

This tests whether the value of $NewSchemeName is unique or not, and can be used to create a new provisioning scheme or rename an existing one without failing. True is returned if the name is good.

```
Test-ProvSchemeNameAvailable -ProvisioningSchemeName $NewSchemeName

Name      : NewScheme1
Available : True
```

Parameters

-ProvisioningSchemeName

The name or names of the provisioning scheme(s) to be tested.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Test-ProvSchemeNameAvailable.

Outputs

Citrix.MachineCreation.Sdk.NameAvailability

Key value pairs of name and its availability.

Name <string>

The provisioning scheme name to be tested.

Available <bool>

The availability of the name.

Notes

In the case of failure, the following errors can result.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- New-ProvScheme
- Rename-ProvScheme

Test-ProvSetProvScheme

March 11, 2024

Test the parameters that can be used for updating provisioning scheme with Set-ProvScheme command

Syntax

```
Test-ProvSetProvScheme
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvSetProvScheme
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolName <String>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvSetProvScheme
    [-ProvisioningSchemeName] <String>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```



```
Test-ProvSetProvScheme
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvSetProvScheme
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolName <String>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-ProvSetProvScheme
    -ProvisioningSchemeUid <Guid>
    [-VMCpuCount <Int32>]
    [-VMMemoryMB <Int32>]
    -IdentityPoolUid <Guid>
    [-CustomProperties <String>]
    [-ServiceOffering <String>]
    [-PassThru]
    [-MachineProfile <String>]
    [-NetworkMapping <Hashtable>]
    [-SecurityGroup <String[]>]
    [-WriteBackCacheDiskSize <Int32>]
    [-WriteBackCacheMemorySize <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```


Description

Provides the ability to validate the parameters of an existing provisioning scheme without actually updating the provisioning scheme. It will output all the validation failures if any.

The following parameters can be tested:

- Number of CPUs that will be used for VMs created from the provisioning scheme.
- Maximum amount of memory that will be used for VMs created from the provisioning scheme.
- Identity pool that will be used for VMs created from the provisioning scheme.
- Machine profile that will be used for VMs created from the provisioning scheme.
- Cloud service offering that will be used for VMs created from the provisioning scheme.
- Custom properties of the provisioning scheme that are specific to the target hosting infrastructure.

Examples

EXAMPLE 1

Validates the given provisioning scheme parameters

```
1  Test-ProvSetProvScheme -ProvisioningSchemeName MyScheme -
      WriteBackCacheDiskSize 50 -VMCpuCount 50 -VMMemoryMB 1000
2  Parameters               ErrorId
                                                  ErrorCategory
                    ErrorMessage
3  ----------               -------
                                               -------------
      ------------
4  {
5   WriteBackCacheDiskSize }
6    WriteBackCacheParametersNotNeeded
        McsValidationErrorCategory WriteBackCacheDiskSize is not needed
        when write back cache is disabled.
7  {
8   CpuCount, MemoryMB }
9        CpuAndMemorySettingViaServiceOfferingOnly
            McsValidationErrorCategory Hypervisor does not support setting
             cpu or memory directly, they can only be specified via
            ServiceOffering.
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme to be updated.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolName

The name of the identity pool to be used for the provisioning scheme, replacing the present one.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-IdentityPoolUid

The unique identifier of the identity pool to be used for the provisioning scheme, replacing the present one.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProvisioningSchemeUid

The identifier of the provisioning scheme to be updated.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMCpuCount

The number of processors that will be used to create VMs from the provisioning scheme, replacing the present one.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-VMMemoryMB

The maximum amount of memory that will be used to created VMs from the provisioning scheme in MB, replacing the present one.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-CustomProperties

The properties of the provisioning scheme that are specific to the target hosting infrastructure. See about_ProvCustomProperties for more information. If a property name already exists its value is updated; otherwise it is added.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceOffering

The Service Offering to use when creating VMs in Cloud Hypervisors, replacing the present one.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named

Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineProfile

Currently only supported with Azure and AWS. Defines the inventory path to the source VM used by the provisioning scheme as a template. This profile identifies the properties for the VMs created from the scheme. The VM must be in the hosting unit that HostingUnitName or HostingUnitUid refers to. If any properties are present in the MachineProfile but not the CustomProperties, values from the template will be written back to the CustomProperties.

Valid paths are of the format: XDHyp:\HostingUnits\<HostingUnitName>\<path>\<VMName>.vm

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-NetworkMapping

Specifies how the attached NICs are mapped to networks. If this parameter is omitted, the current NICs setting of the provisioning scheme is not updated; otherwise the NICs setting is updated, and new machines will be created with the number of NICs specified in the map, with each NIC attached to the specified network. Cannot set to an empty value.

Type:	Hashtable
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-SecurityGroup

The Security Groups to use when creating VMs in Cloud Hypervisors, replacing the present one.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheDiskSize

Specifies the size in Gigabytes of the disk to use as a Write Back Cache when UseWriteBackCache is set during the Provisioning Scheme creation. This parameter can only be set or modified if the Provisioning Scheme was previously created with UseWriteBackCache. This parameter only applies to newly created VMs and does not affect VMs which have already been created from the Provisioning Scheme.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-WriteBackCacheMemorySize

Specifies the size in Megabytes of the memory to use as a Write Back Cache when UseWriteBackCache is set during the Provisioning Scheme creation. This parameter can only be set or modified if the Provisioning Scheme was previously created with UseWriteBackCache. This parameter only applies

to newly created VMs and does not affect VMs which have already been created from the Provisioning Scheme. Setting this parameter to 0 disables the use of memory for Write Back Cache.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

IList<Citrix.MachineCreation.Sdk.ProvSchemeValidationFailure>

This object provides a list of validation error messages for given provisioning scheme parameters and contains the following information:

Parameters <IList<string» Name of the parameter that failed validation

ErrorId <string> Error code

ErrorCategory <string> The source of error - MCS or Plugin

ErrorMessage <string> Friendly message describing the validation failure

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound The specified provisioning scheme could not be located.

DatabaseError An error occurred in the service while attempting a database operation.

DatabaseNotConfigured The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb An error occurred in the service while attempting a database operation - communication with the database failed for for various reasons.

CommunicationError An error occurred while communicating with the service.

PermissionDenied The user does not have administrative rights to perform this operation.

ConfigurationLoggingError The operation could not be performed because of a configuration logging error.

HostingUnitNotFound The hosting unit referenced by the provisioning scheme could not be resolved.

NetworkPathResolutionFailed The specified network mapping contains invalid or empty network path value.

WriteBackCacheParametersNotNeeded WriteBackCacheDiskSize is not needed when write back cache is disabled.

CpuAndMemorySettingViaServiceOfferingOnly Hypervisor does not support setting cpu or memory directly, they can only be specified via ServiceOffering.

NetworkPathResolutionFailed The specified network path could not be resolved. Please ensure that the path includes a drive specification and path to a location within a HostingUnit.

InvalidMachineProfileMetadata Unable to read the metadata from machine profile.

MachineProfileNotSupportedByHypervisor The hypervisor does not support any functional capability for MachineProfile.

MachineProfileUpdateNotSupported Adding a MachineProfile to the catalog is not supported.

IncorrectTenancyType MachineProfile TenancyType does not match with provisioning scheme.

ServiceOfferingPathResolutionFailed The Service Offering path could not be resolved. Please ensure that the path includes a drive specification and path to a location within a HostingUnit.

SecurityGroupPathResolutionFailed The Security Group path could not be resolved. Please ensure that the path includes a drive specification and path to a location within a HostingUnit.

ExceptionThrown An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_Prov_CustomProperties
- Set-ProvScheme

Unlock-ProvScheme

March 11, 2024

Unlocks a provisioning scheme.

Syntax

```
Unlock-ProvScheme
       [-ProvisioningSchemeName] <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

```
Unlock-ProvScheme
       -ProvisioningSchemeUid <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to unlock a provisioning scheme that has the Id of a failed task still associated with it. This allows another long-running task to operate on that scheme. The cmdlet will not unlock a scheme with a task still marked as being active. Use %5BStop-ProvTask%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Stop-ProvTask.html) (or, if the task is registered with a failed server, Switch-ProvTask and %5BStop-ProvTask%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/MachineCreation/Stop-ProvTask.html)) to stop any active task first.

Examples

EXAMPLE 1

Unlocks the provisioning scheme 'MyScheme'.

```
Unlock-ProvScheme -provisioningSchemeName MyScheme
```

Parameters

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Unlock-ProvScheme.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

NoOp

The specified provisioning scheme had no task object associated.

TaskActive

The task object associated with the provisioning scheme is still active.

DatabaseError

An error occurred in the service while attempting a database operation.

CouldNotQueueryDatabase

The query required to get the database was not defined.

CommunicationError

An error occurred while communicating with the service.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvScheme
- Stop-ProvTask

Unlock-ProvVM

March 11, 2024

Unlocks a virtual machine.

Syntax

```
Unlock-ProvVM
       -ProvisioningSchemeName <String>
       [-VMID] <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```



```
Unlock-ProvVM
       -ProvisioningSchemeUid <Guid>
       [-VMID] <String[]>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to unlock a virtual machine.

Examples

EXAMPLE 1

Unlocks the VM with the Id 'bc79802c-ba6e-8de8-99e9-4c35d7ad24b4'in the provisioning scheme 'MyScheme'.

```
Unlock-ProvVM -provisioningSchemeName MyScheme -VMId  bc79802c-ba6e-8
    de8-99e9-4c35d7ad24b4
```

EXAMPLE 2

Unlocks all the VMs in the provisioning scheme 'MyScheme'.

```
Get-ProvVM -provisioningSchemeName MyScheme | Unlock-ProvVM
```

EXAMPLE 3

Unlocks all the VMs that are currently locked.

```
Get-ProvVM -Locked $True | Unlock-ProvVM
```

Parameters

-VMID

The virtual machine Id in the hypervisor.

Type: String[]

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeName

The name of the provisioning scheme.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-ProvisioningSchemeUid

The unique identifier of the provisioning scheme.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisionedVirtualMachine

You can pipe an object containing parameters called 'VMId' and 'ProvisioningSchemeName' to Unlock-ProvVM.

Outputs

None

By default, this cmdlet returns no output.

Notes

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

VMDoesNotExist

The specified VM cannot be located.

VMAlreadyUnLocked

The VM is already unlocked.

VMDoesNotExistForProvisioningScheme

The specified VM does exist in the hypervisor, but is not part of the specified provisioning scheme.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for

for various reasons.

CommunicationError

An error occurred while communicating with the service.

PermissionDenied

The user does not have administrative rights to perform this operation.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller being used or Citrix Virtual Apps and Desktops logs.

Related Links

- about_ProvMachineCreationSnapin
- Get-ProvVM
- Lock-ProvVM

Unpublish-ProvMasterVMImage

March 11, 2024

Remove a prepared master image from a provisioning scheme.

Syntax

```
Unpublish-ProvMasterVMImage
          [-ProvisioningSchemeName] <String>
          [-LoggingId <Guid>]
          [<CitrixCommonParameters>]
          [<CommonParameters>]
```

```
Unpublish-ProvMasterVMImage
          -ProvisioningSchemeUid <Guid>
          [-LoggingId <Guid>]
          [<CitrixCommonParameters>]
          [<CommonParameters>]
```

```
Unpublish-ProvMasterVMImage
          -VMImageHistoryUid <Guid>
          [-LoggingId <Guid>]
          [<CitrixCommonParameters>]
          [<CommonParameters>]
```

Description

Provides the ability to remove a master image and its hard disk copies that has been prepared for a provisioning scheme to create virtual machines.

The master image is removed from the image history of the provisioning scheme (see Get-ProvSchemeMasterVMImageHistory).

A background task will be created to remove the old hard disk copies. You can locate and monitor this task using the Get-ProvTask cmdlet.

Examples

EXAMPLE 1

Removes the prepared master images for the provisioning Scheme "MyScheme".

```
1 Unpublish-ProvMasterVMImage -ProvisioningSchemeName MyScheme
```

EXAMPLE 2

Removes a prepared image identified by its history record.

```
1 Unpublish-ProvMasterVMImage -VMImageHistoryUid 7585f0de-192e-4847-a6d8
    -22713c3a2f42 -Revoke
```

Parameters

-ProvisioningSchemeName

Name of the provisioning scheme from which the specified master image should be removed.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ProvisioningSchemeUid

Unique identifier of the provisioning scheme from which the specified master image should be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-VMImageHistoryUid

Unique identifier of the master image to be removed.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.MachineCreation.Sdk.ProvisioningScheme

You can pipe an object containing a parameter called 'ProvisioningSchemeName'to Unpublish-ProvMasterVMImage.

Outputs

None

By default, this cmdlet returns no output.

Notes

If a VM Image History UID is specified, the cmdlet will remove the image if it is in the prepared state. If the image is the current image of the provisioning scheme, an error will be returned: there must be a current image associated with a provisioning scheme. The Publish-ProvMasterVMImage cmdlet may be used to replace a current image with a new or previously prepared image.

If a provisioning scheme name or UID is specified, the cmdlet will remove the prepared master image associated with that provisioning scheme if it exists. No error will be returned if no prepared image exists. The current image of the scheme will not be removed.

In the case of failure, the following errors can result.

Error Codes

ProvisioningSchemeNotFound

The specified provisioning scheme could not be located.

ProvisioningSchemeNotReady

The specified provisioning scheme is not in Ready state.

Related Links

- about_ProvMachineCreationSnapin
- Publish-ProvMasterVMImage
- Get-ProvSchemeMasterVMImageHistory
- Get-ProvScheme

about_MonitorMonitorSnapIn

March 11, 2024

Topic

about_MonitorMonitorSnapin

Short Description

The Monitor Service PowerShell snap-in provides administrative functions for the Monitor Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Monitor'.

Long Description

The Monitor Service PowerShell snap-in enables both local and remote administration of the Monitor service. It provides
facilities to query the Monitor service configuration settings and to modify those settings. It also provides the
standard set of Citrix Virtual Apps and Desktops 7 service cmdlets.

about_MonitorMonitorSnapIn

March 11, 2024

Topic

about_MonitorMonitorSnapin

Short Description

The Monitor Service PowerShell snap-in provides administrative functions for the Monitor Service.

Command Prefix

All commands in this snap-in have the noun prefixed with 'Monitor'.

Long Description

The Monitor Service PowerShell snap-in enables both local and remote administration of the Monitor service. It provides
facilities to query the Monitor service configuration settings and to modify those settings. It also provides the
standard set of Citrix Virtual Apps and Desktops 7 service cmdlets.

about_Monitor_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also

search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z"}
$d = [DateTime]"2010-08-23T12:30:00.0Z"

Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2'} # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30'} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30'} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*"}
Get-<Noun> -Filter { Users -contains "Fred*"}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'

Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects

with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries. It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-MonitorNotificationPolicyCondition

March 11, 2024

Add conditions to the existing policy specified and returns the updated policy.

Syntax

```
Add-MonitorNotificationPolicyCondition
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyCondition
   -InputObject <MonitorNotificationPolicy>
   [-AlertThreshold <Int32>]
   [-AlarmThreshold <Int32>]
   [-AlertRenotification <TimeSpan>]
   [-AlarmRenotification <TimeSpan>]
   [-SearchWindow <TimeSpan>]
   [-AlertConditionPersistenceInterval <TimeSpan>]
   [-AlarmConditionPersistenceInterval <TimeSpan>]
   [-Granularity <TimeSpan>]
   [-AlertSampleCount <Int32>]
   [-AlarmSampleCount <Int32>]
   [-AlertSamplePercent <Int32>]
   [-AlarmSamplePercent <Int32>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyCondition
   -Uid <Int64>
   -ConditionType <ConditionType>
   [-AlertThreshold <Int32>]
   [-AlarmThreshold <Int32>]
   [-AlertRenotification <TimeSpan>]
   [-AlarmRenotification <TimeSpan>]
   [-SearchWindow <TimeSpan>]
   [-AlertConditionPersistenceInterval <TimeSpan>]
   [-AlarmConditionPersistenceInterval <TimeSpan>]
   [-Granularity <TimeSpan>]
   [-AlertSampleCount <Int32>]
```



```
    [-AlarmSampleCount <Int32>]
    [-AlertSamplePercent <Int32>]
    [-AlarmSamplePercent <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Add conditions to the existing policy specified and returns the updated policy.

Examples

EXAMPLE 1

Add SessionsConcurrentCount condition to the policy matching the id 100

```
$timeSpan = New-TimeSpan -Seconds 30
$alertThreshold = 10
$alarmThreshold = 20

Add-MonitorNotificationPolicyCondition -Uid 100 -ConditionType
    SessionsConcurrentCount -AlertThreshold $alertThreshold -
    AlarmThreshold $alarmThreshold  -AlertRenotification $timeSpan -
    AlarmRenotification $timeSpan
```

Parameters

-InputObject

Specifies the policy object to which the conditions to be added

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy to which the conditions to be added

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ConditionType

Type of the condition using a text string converted to an enum value Possible values are SessionsConcurrentCount SessionsPeakconnectedCount SessionsPeakDisconnectedCount AverageLogonDuration RDSLoadEvaluator ConnectionFailuresRate ConnectionFailuresCount FailedServerMachineCount FailedDesktopMachineCount LogonDuration IcaRoundtripTime IcaRoundtripTimeAverage IcaRoundtripTimeSessionCount IcaRoundtripTimeSessionPercent MemoryUsage CpuUsage FailedMachinePercentage

Type:	ConditionType
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AlertThreshold

Threshold value at which the warning notification will be triggered

Type:	Int32
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlarmThreshold

Threshold value at which the critical notification will be triggered

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlertRenotification

Duration after which the warning notification will be re-triggered

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlarmRenotification

Duration after which the critical notification will be re-triggered

Type:	TimeSpan
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SearchWindow

The amount of time the condition query will look back for the matching record when it examines the data.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlertConditionPersistenceInterval

The amount of time the condition query will look back to check if the alert condition was persisted.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlarmConditionPersistenceInterval

The amount of time the condition query will look back to check if the alarm condition was persisted.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Granularity

Defines the granularity of the rate in seconds. This value is applicable for 'ConnectionFailuresRate' condition.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlertSampleCount

Defines the count of instances to measure before raising the warning notification. This value is applicable for ICA RTT (Session Count) condition.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlarmSampleCount

Defines the count of instances to measure before raising the critical notification. This value is applicable for ICA RTT (Session Count) condition.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlertSamplePercent

Defines the percentage of instances to measure before raising the warning notification. This value is applicable for ICA RTT (Session Percent) condition.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AlarmSamplePercent

Defines the percentage of instances to measure before raising the critical notification. This value is applicable for ICA RTT (Session Percent) condition.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the updated policy object

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Add-MonitorNotificationPolicyEmailAddresses

March 11, 2024

Add email addresses to the existing policy specified and returns the updated policy.

Syntax

```
Add-MonitorNotificationPolicyEmailAddresses
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyEmailAddresses
   -InputObject <MonitorNotificationPolicy>
   -EmailAddresses <String[]>
   -EmailCultureName <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyEmailAddresses
   -Uid <Int64>
   -EmailAddresses <String[]>
   -EmailCultureName <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Add email addresses to the existing policy specified and returns the updated policy.

Examples

EXAMPLE 1

Add email addresses to the policy matching the id 100

```
$emailaddress = @()
$emailaddress += "user1@abc.com"

Add-MonitorNotificationPolicyEmailAddresses -Uid 100 -EmailAddresses
    $emailaddress -EmailCultureName "en-US"
```

Parameters

-InputObject

Specifies the policy object to which the conditions to be added.

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-EmailAddresses

Collection of email addresses to be added

Type:	String[]
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-EmailCultureName

Text encoding of the culture to return the email in

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy to which the conditions to be added.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Collection[MonitorNotificationPolicy]

Returns the collection of policy objects created

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Add-MonitorNotificationPolicyTargets

March 11, 2024

Add targets to the existing policy specified and returns the updated policy

Syntax

```
Add-MonitorNotificationPolicyTargets
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyTargets
   -InputObject <MonitorNotificationPolicy>
   [-Scope <String>]
   [-TargetKind <TargetKind>]
   -TargetIds <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Add-MonitorNotificationPolicyTargets
   -Uid <Int64>
   [-Scope <String>]
   [-TargetKind <TargetKind>]
   -TargetIds <String[]>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Add targets to the existing policy specified and returns the updated policy

Examples

EXAMPLE 1

Add targets to the policy matching the id 100

```
$targetIds = @()
$targetIds += "766cde70-3c69-4481-a658-4e11247ac70c"

```



```
4 Add-MonitorNotificationPolicyTargets -Uid 100 -Scope "My Test Targets"
      -TargetKind Site -TargetIds $targetIds
```

Parameters

-InputObject

Specifies the policy object to which the targets to be added

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-TargetIds

Object reference to the array of target ids.

Site GUID - for target type Site,

DesktopGroup UUID - for DesktopGroup, RdsWorker target types,

User SID - for User target type

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy to which the targets to be added

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Scope

Description of the collection of target ids

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetKind

String representation of the target type enum.

Possible values are

Site

DesktopGroup

RdsWorker

User

Type:	TargetKind
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the policy object

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Get-MonitorConfiguration

March 11, 2024

Gets configuration settings that are used by the Monitor Service.

Syntax

```
Get-MonitorConfiguration
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the configuration settings used by the Monitor Service. These settings are used to modify the behavior of the service.

Examples

EXAMPLE 1

Get the settings in the site database.

```
Get-MonitorConfiguration
```


Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

String

List of configuration settings used by the Monitor Service.

Related Links

- Set-MonitorConfiguration

Get-MonitorDataStore

March 11, 2024

Gets details for each of the Monitor data stores.

Syntax

```
Get-MonitorDataStore
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns an object for each of the Monitor data stores describing the connection string, data store name, db type, provider, schema name, and DB status.

A database connection must be configured in order for this command to be used if the service has a secondary data store. This is not required for the site data store.

Examples

EXAMPLE 1

Get the database connection string for the Monitor Service.

```
Get-MonitorDataStore

ConnectionString : Server=.\SQLEXPRESS;Initial Catalog = databaseName;
   Integrated Security = True
DataStore        : Site
DatabaseType     : SqlServer
Provider         : MSSQL
SchemaName       : MonitorSiteSchema
Status           : OK

ConnectionString :
DataStore        : Secondary
```



```
DatabaseType      : SqlServer
Provider          : MSSQL
SchemaName        : MonitorSecondarySchema
Status            : DBUnconfigured
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Monitor.Sdk.DataStoreConfiguration

An object describing the connection string, data store name, database type, provider, schema name and database status.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops 7 logs.

Related Links

- about_MonitorMonitorSnapin
- Reset-MonitorDataStore

Get-MonitorDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Monitor Service.

Syntax

```
Get-MonitorDBConnection
   [[-DataStore] <String>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets the database connection string from the currently selected Monitor Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Monitor Service instance running on controller "controller1.mydomain.net".

```
1  Get-MonitorDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	2
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Monitor Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the MonitorService has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Set-MonitorDBConnection
- Get-MonitorServiceStatus
- Test-MonitorDBConnection
- Get-MonitorDataStore

Get-MonitorDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Monitor Service.

Syntax

```
Get-MonitorDBSchema
    [-DataStore <String>]
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Monitor Service database schema, add a new Monitor service to an existing site, remove a Monitor service from a site, or create a database server logon for a Monitor service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Monitor service instance, otherwise the scripts relate to Monitor service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Monitor service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Monitor service roles

If ScriptType is Evict, the returned script contains:

- Removal of Monitor service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Monitor Service and copies it to a file called "C:\MonitorSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Monitor service schema.

```
Get-MonitorDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\MonitorSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Monitor service. This can be used when configuring a mirror server for use.

```
Get-MonitorDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    MonitorLogins.sql
```

EXAMPLE 3

Get the full database schema for the secondary data store of the Monitor Service and copy it to a file called 'c:\MonitorSecondarySchema.sql'.

This script can then be used to create the schema in a pre-existing database named 'MyDB'that does not already contain a Monitor Service secondary schema.

```
Get-MonitorDBSchema -DatabaseName MyDB -ServiceGroupName MyServiceGroup
     -DataStore Secondary > c:\MonitorSchema.sql
```

Parameters

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site'or the logical name of the secondary data store.

Type:	String
Position:	Named

Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

Specifies the name of the database into which the new Monitor service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Monitor services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Monitor Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Monitor Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Monitor Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Monitor Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the

required permissions for local services to access the database schema for Monitor services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Monitor Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorDataStore
- Set-MonitorDBConnection
- Test-MonitorDBConnection

Get-MonitorDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Monitor Service.

Syntax

```
Get-MonitorDBVersionChangeScript
    [-DataStore <String>]
    -DatabaseName <String>
    -TargetVersion <Version>
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Monitor Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Monitor Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Monitor Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Monitor Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Monitor Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-MonitorServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-MonitorDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Monitor Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	Named
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-MonitorDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Monitor Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Monitor services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-MonitorServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorInstalledDBVersion
- Get-MonitorServiceStatus
- Get-MonitorDBSchema

Get-MonitorInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Monitor Service.

Syntax

```
Get-MonitorInstalledDBVersion
   [-DataStore <String>]
   [-Upgrade]
   [-Downgrade]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Monitor Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-MonitorDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Monitor Service database schema version number.

```
Get-MonitorInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Monitor Service database schema for which upgrade scripts are supplied.

```
Get-MonitorInstalledDBVersion -Upgrade
```

Parameters

-DataStore

Specifies the database connection logical name the schema script should be returned for. The parameter is optional.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-MonitorInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Monitor

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorDBVersionChangeScript
- Get-MonitorDBSchema

Get-MonitorNotificationEmailServerConfiguration

March 11, 2024

Gets email server configuration

Syntax

```
Get-MonitorNotificationEmailServerConfiguration
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrurns the existing email server configurations required to send email alerts.

Examples

EXAMPLE 1

Returns the email configuration

```
GetMonitorNotificationEmailServerConfigurationCommand
```


Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

ICollection<MonitorNotificationEmailServerConfiguration> System.String

Email server configuration

Related Links

- Set-MonitorNotificationEmailServerConfiguration

Get-MonitorNotificationPolicy

March 11, 2024

Returns a list of MonitorNotificationPolicy objects matching the specified parameters

Syntax

```
Get-MonitorNotificationPolicy
   [-Uid <Int64>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Get-MonitorNotificationPolicy
   -Name <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a list of MonitorNotificationPolicy objects matching the specified parameters

Examples

EXAMPLE 1

Returns the policy object having the Id equals to 1

```
Get-MonitorNotificationPolicy -Id 1
```

EXAMPLE 2

Returns the policy object having the name contains 'Server OS Policy'

```
Get-MonitorNotificationPolicy -Name 'Server OS Policy'
```


Parameters

-Name

Returns all of the policies contains the specified name.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-Uid

Returns the policy with the specified id.

Type:	Int64
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.ICollection`1[[Citrix.Monitor.Sdk.PowerShell.MonitorNotificationPolicy, Citrix.Monitor.PowerShellSnapIn, Version=7.7.0.95, Culture=neutral, PublicKeyToken=e6a2b8abcb991548]]

Returns a collection of policies.

Related Links

- New-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy

Get-MonitorNotificationSnmpServerConfiguration

March 11, 2024

Gets email server configuration

Syntax

```
Get-MonitorNotificationSnmpServerConfiguration
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns the existing Snmpserver configurations required to send Snmp alerts.

Examples

EXAMPLE 1

Returns the Snmp configuration

```
Get-MonitorNotificationSnmpServerConfiguration
```

Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

ICollection<MonitorNotificationEmailServerConfiguration> System.String

Snmp server configuration

Related Links

- Set-MonitorNotificationSnmpServerConfiguration

Get-MonitorService

March 11, 2024

Gets the service record entries for the Monitor Service.

Syntax

```
Get-MonitorService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the Monitor Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Monitor Service running in the current service group.

```
Get-MonitorService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```


Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*"matches records with a metadata entry having a key name of "abc"and a value starting with the letter "x".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Monitor_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Monitor_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Monitor.Sdk.Service

The Get-MonitorServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin

Get-MonitorServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Monitor Service on the controller.

Syntax

```
Get-MonitorServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Monitor Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Monitor Service.

```
Get-MonitorServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin

Get-MonitorServiceInstance

March 11, 2024

Gets the service instance entries for the Monitor Service.

Syntax

```
Get-MonitorServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Monitor Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Monitor Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-MonitorServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Monitor.Sdk.ServiceInstance

The Get-MonitorServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Monitor.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Monitor.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorServiceStatus
- Reset-MonitorServiceGroupMembership

Get-MonitorServiceStatus

March 11, 2024

Gets the current state of the Monitor Service on the controller.

Syntax

```
Get-MonitorServiceStatus
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables the status of the Monitor Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will

return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Monitor Service.

```
Get-MonitorServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-MonitorServiceStatus command returns an object containing the status of the Monitor Service together with extra diagnostics information.

DBUnconfigured

The Monitor Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Monitor Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Monitor Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Monitor is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Monitor is connected to a database that is valid, but it does not have the full functionality required so the Monitor cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Monitor Service currently in use is incompatible with the version of the Monitor Service schema on the database. Upgrade the Monitor Service to a more recent version.

DBOlderVersionThanService

The version of the Monitor Service schema on the database is incompatible with the version of the Monitor Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Monitor Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Monitor Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Monitor and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Monitor service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorDataStore
- Set-MonitorDBConnection
- Test-MonitorDBConnection
- Get-MonitorDBConnection
- Get-MonitorDBSchema

New-MonitorCssEventNotification

March 11, 2024

Sends a ConfigSyncService failure event notification to Monitor Service.

Syntax

```
New-MonitorCssEventNotification
   -Mode <Int32>
   -Level <Int32>
   -ConsecutiveDbImportFailCount <Int32>
   -SourceMachineName <String>
   -ResourceLocId <Guid>
   -NotificationTime <DateTime>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Sends a ConfigSyncService failure event notification to Monitor Service. These cmdlet is called by the CSS service.

Examples

EXAMPLE 1

Inserts the CSS event notification to NotificationCssEvent records. Updates the level in case there is a change in it.

```
New-MonitorCssEventNotification -AdminAddress "W2K19ST-VN9BLQD.sgs.
    local:89" -Mode 0 -Level 1 -ConsecutiveDbImportFailCount 4 -
    SourceMachineName "W2K19ST-VN9BLQD.sgs.local" -ResourceLocId $(New-
    Guid) -NotificationTime $(Get-Date)
```

Parameters

-Mode

Mode of the event LHC Outage or CSS failure

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Level

Level warning or critical

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ConsecutiveDbImportFailCount

Count of the consecutive DB import failure on Connector/Controller machine

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SourceMachineName

Connector or notification source machine name

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ResourceLocId

Resource location Id of the notification source

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-NotificationTime

Time the event sent from ConfigSyncService

Type:	DateTime
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Status of the setting a new CSS failure event notification

Related Links

New-MonitorLhcReports

March 11, 2024

Returns a boolean

Syntax

```
New-MonitorLhcReports
   -EndDate <DateTime>
   -Duration <Single>
   -MachineInZoneCount <Int32>
   -MachineRegisteredCount <Int32>
   -ReconnectedSessionCount <Int32>
   -SessionBrokeredCount <Int32>
   -ZoneName <String>
   -ControllerName <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a boolean stating whether a new lhc report was added or not

Examples

EXAMPLE 1

Inserts multiple attributes into LhcReport after the cloud connector outage is over

```
New-MonitorLhcReports -Duration 120 -EndDate 2023-09-05T09
    :13:39.1835337Z -MachineInZoneCount 1 -MachineRegisteredCount 2 -
    SessionBrokeredCount 3 -ReconnectedSessionCount 4 -ZoneName "
    Bangalore" -ControllerName "Dynasty"
```

Parameters

-EndDate

End date of LHC

Type:	DateTime
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Duration

Duration of LHC in minutes

Type:	Single
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineInZoneCount

Count of Machines in zone

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MachineRegisteredCount

Count of machines registered

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ReconnectedSessionCount

Count of sessions reconnected

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionBrokeredCount

Count of sessions brokered

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ZoneName

Name of zone

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ControllerName

Name of Controller/Cloud connector

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Returns a bool stating LhcReport was saved into db

Related Links

New-MonitorNotificationPolicy

March 11, 2024

Returns a new MonitorNotificationPolicy object

Syntax

```
New-MonitorNotificationPolicy
   -Name <String>
   [-Description <String>]
   [-Webhook <String>]
   [-IsSnmpEnabled <Boolean>]
   [-Enabled <Boolean>]
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns a new policy instance using the specified parameters

Examples

EXAMPLE 1

Creates session policy and persist the data into database

```
New-MonitorNotificationPolicy -Name "Policy1" -Description "Policy
    Description" -Enabled $true
```

Parameters

-Name

Name of policy to create

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

Description of policy to create

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Webhook

Webhook URL of policy to create

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsSnmpEnabled

boolean value representing if SNMP is enabled for new Policy

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean paramter indicating the enabled state of the policy. true - Enabled, false - Disabled

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the collection of policy objects created

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Remove-MonitorNotificationPolicy

March 11, 2024

Removes persisted policy from the database by marking them as deleted

Syntax

```
Remove-MonitorNotificationPolicy
      [-InputObject <MonitorNotificationPolicy[]>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorNotificationPolicy
      [-Uid <Int64[]>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Removes the policy object from database by marking them as deleted and disable all the condtions associated with that

Examples

EXAMPLE 1

Removes the policy with id 1

```
Remove-MonitorNotificationPolicy -Id 1
```

EXAMPLE 2

Removes the policy with id matching the policy object specified

```
Remove-MonitorNotificationPolicy -InputObject $policy
```

Parameters

-InputObject

Removes all of the instances with the id matching in the specified policy objects

Type:	MonitorNotificationPolicy[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Uid

Removes all of the instances with the specified ids. If any of the ids are invalid, an exception is thrown.

Type:	Int64[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Return success if all operations are succeeded

Return success if all operations are succeeded. An exception is thrown otherwise

Related Links

- New-MonitorNotificationPolicy
- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy

Remove-MonitorNotificationPolicyConditions

March 11, 2024

Remove conditions from the existing policy specified and returns the updated policy.

Syntax

```
Remove-MonitorNotificationPolicyConditions
      -InputObject <MonitorNotificationPolicy>
      -Conditions <ConditionType[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorNotificationPolicyConditions
      -Uid <Int64>
      -Conditions <ConditionType[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Remove conditions from the existing policy specified and returns the updated policy.

Examples

EXAMPLE 1

Removes the condition SessionPeakconnectedCount from policy matching id 100

```
Remove-MonitorNotificationPolicyConditions -Uid 100 -Conditions
    SessionsPeakconnectedCount
```

Parameters

-InputObject

Specifies the policy object from which the conditions to be removed.

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Conditions

Collection of condition types to be removed

Type:	ConditionType[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy from which the conditions to be removed.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the updated policy object

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Remove-MonitorNotificationPolicyEmailAddresses

March 11, 2024

Remove email addresses from the existing policy specified and returns the updated policy.

Syntax

```
Remove-MonitorNotificationPolicyEmailAddresses
      -InputObject <MonitorNotificationPolicy>
      -EmailAddresses <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorNotificationPolicyEmailAddresses
      -Uid <Int64>
      -EmailAddresses <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Remove email addresses from the existing policy specified and returns the updated policy.

Examples

EXAMPLE 1

Removes the email addresses from the policy matching id 100

```
Remove-MonitorNotificationPolicyEmailAddresses -Uid 100 -EmailAddresses
    $emailAddresses
```

Parameters

-InputObject

Specifies the policy object from which the email addresses to be removed

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-EmailAddresses

Collection of email addresses to be removed

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy from which the email addresses to be removed.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the updated policy object

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Remove-MonitorNotificationPolicyTargets

March 11, 2024

Remove targets from the existing policy specified and returns the updated policy.

Syntax

```
Remove-MonitorNotificationPolicyTargets
      -InputObject <MonitorNotificationPolicy>
      -TargetIds <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorNotificationPolicyTargets
      -Uid <Int64>
      -TargetIds <String[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Remove targets from the existing policy specified and returns the updated policy.

Examples

EXAMPLE 1

Removes the targets from policy matching id 100

```
$targetIds = @()
$targetIds += "766cde70-3c69-4481-a658-4e11247ac70d"

Remove-MonitorNotificationPolicyTargets -Uid 100  -TargetIds $targetIds
```

Parameters

-InputObject

Specifies the policy object from the targets to be removed

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-TargetIds

Object reference to the array of target ids.

Site GUID - for target type Site

DesktopGroup UUID - for DesktopGroup, RdsWorker target types

Type:	String[]
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Specifies the unique identifier of the policy from the targets to be removed

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the policy object

Related Links

- Get-MonitorNotificationPolicy
- Set-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Remove-MonitorServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-MonitorServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-MonitorServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-MonitorService | % {
  Remove-MonitorServiceMetadata -Map $_.MetadataMap  }
```


Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Set-MonitorServiceMetadata

Reset-MonitorDataStore

March 11, 2024

Refreshes the database string currently being used by the Monitor service.

Syntax

```
Reset-MonitorDataStore
      [-DataStore] <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Returns the string for the database connection currently being used by the Monitor Service. Can only be called for secondary data stores.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Refresh the database connection string for the Monitor Service.

```
Reset-MonitorDataStore -DataStore Secondary
OK
```

Parameters

-DataStore

Specifies the database connection logical name to be used by the Monitor Service. Can be either be 'Site'or the logical name of the secondary data store. Specifying the site data store will display an error because this operation is not supported for site data stores.

Type:	String
Position:	2
Default value:	Site
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Monitor.Sdk.ServiceStatus

The status of the specified data store.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the Citrix Virtual Apps and Desktops 7 logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorDataStore

Reset-MonitorEnabledFeatureList

March 11, 2024

Refreshes the Monitor service's list of enabled features.

Syntax

```
Reset-MonitorEnabledFeatureList
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Monitor Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Monitor service instance's list of enabled features.

```
Reset-MonitorEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-MonitorServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Monitor Service.

Syntax

```
Reset-MonitorServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Monitor Service access permissions and configuration service locations. The Reset-MonitorServiceGroupMembership command must be run on at least one instance of the service type (Monitor) after installation and registration with the configuration service. Without this operation, the Monitor services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-MonitorServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    MonitorServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-MonitorServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Monitor.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-MonitorServiceGroupMembership command.

Outputs

Citrix.Monitor.Sdk.ServiceInstance

Reset-MonitorServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Monitor Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorServiceInstance
- Get-MonitorServiceStatus

Set-MonitorConfiguration

March 11, 2024

Sets configuration settings that are used by the Monitor Service.

Syntax

```
Set-MonitorConfiguration
    [-GroomSessionsRetentionDays <Int32>]
    [-GroomFailuresRetentionDays <Int32>]
    [-GroomLoadIndexesRetentionDays <Int32>]
    [-GroomDeletedRetentionDays <Int32>]
    [-GroomSummariesRetentionDays <Int32>]
    [-GroomMachineHotfixLogRetentionDays <Int32>]
    [-GroomMinuteRetentionDays <Int32>]
    [-GroomHourlyRetentionDays <Int32>]
    [-DataCollectionEnabled <Boolean>]
    [-FullPollStartHour <Int32>]
    [-GroomStartHour <Int32>]
    [-GroomStartMinuteOffsetEarliest <Int32>]
    [-GroomStartMinuteOffsetLatest <Int32>]
    [-SyncDataToCasStartHour <Int32>]
    [-ResolutionPollTimeHours <Int32>]
    [-SyncPollTimeHours <Int32>]
    [-DetailedSqlOutputEnabled <Boolean>]
    [-MonitorQueryTimeoutSeconds <Int32>]
    [-CollectHotfixDataEnabled <Boolean>]
    [-GroomApplicationInstanceRetentionDays <Int32>]
    [-GroomApplicationErrorsRetentionDays <Int32>]
    [-GroomApplicationFaultsRetentionDays <Int32>]
    [-GroomNotificationLogRetentionDays <Int32>]
    [-GroomResourceUsageRawDataRetentionDays <Int32>]
    [-GroomResourceUsageMinuteDataRetentionDays <Int32>]
    [-GroomResourceUsageHourDataRetentionDays <Int32>]
    [-GroomResourceUsageDayDataRetentionDays <Int32>]
    [-GroomProcessUsageHourDataRetentionDays <Int32>]
    [-GroomProcessUsageDayDataRetentionDays <Int32>]
    [-GroomSessionMetricsDataRetentionDays <Int32>]
    [-GroomMachineMetricDataRetentionDays <Int32>]
    [-GroomMachineMetricDaySummaryDataRetentionDays <Int32>]
    [-EnableDayLevelGranularityProcessUtilization <Boolean>]
    [-GroomApplicationProbeLogsRetentionDays <Int32>]
    [-EnableHourLevelGranularityProcessUtilization <Boolean>]
    [-EnableMinLevelGranularityProcessUtilization <Boolean>]
    [-GroomSessionAutoReconnectsRetentionDays <Int32>]
    [-DisableHypervisorMonitoring <Boolean>]
    [-SendProcessDataToCASAndDatabase <Boolean>]
    [-SendProcessDataToCASAndSkipDatabase <Boolean>]
    [-DisablePendoTrackEvents <Boolean>]
    [-EnableProcessDataLimitedMachines <Boolean>]
    [-SessionRecordingDataProcessing <Boolean>]
    [-GroomProcessDataLimitedMachinesRetentionDays <Int32>]
    [-ProcessDataLimitedMachinesCount <Int32>]
    [-GroomLhcReportRetentionDays <Int32>]
    [-LoggingId <Guid>]
```



```
49      [<CitrixCommonParameters>]
50      [<CommonParameters>]
```

Description

Sets the configuration settings used by the Monitor Service. Use these settings to modify the behavior of the service.

A database connection need not be configured for this command to be used.

Examples

EXAMPLE 1

Updates the settings in the site database with the newly specified values.

```
1  Set-MonitorConfiguration -GroomSessionsRetentionDays 5 -
       GroomFailuresRetentionDays 4 ...
```

Parameters

-GroomSessionsRetentionDays

Determines how many days to keep Session and Connection records after the Session is terminated.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomFailuresRetentionDays

Determines how many days to keep MachineFailureLog and ConnectionFailureLog records after these are created.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomLoadIndexesRetentionDays

Determines how many days to keep LoadIndex records after these are created.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomDeletedRetentionDays

Determines how many days to keep Machine, Catalog, DesktopGroup and Hypervisor entities around that have a LifecycleState of 'Deleted'. This also deletes any related Session, Connection, Summary, Failure or LoadIndex records.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomSummariesRetentionDays

Determines how many days to keep DesktopGroupSummary, FailureLogSummary and LoadIndex-Summary records at the daily granularity.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomMachineHotfixLogRetentionDays

Determines how many days to keep Machine-Hotfix history records at the daily granularity.

Type:	Int32
Position:	Named
Default value:	90
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomMinuteRetentionDays

Determines how many days to keep minute data.

Type:	Int32
Position:	Named
Default value:	3
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomHourlyRetentionDays

Determines how many days to keep hourly data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 32 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataCollectionEnabled

Starts / stops data collection. Stopping data collection turns off polling, and does not persist operational event data to the database.

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FullPollStartHour

Hour of day when Full Poll should begin.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomStartHour

Hour of day when Grooming should begin. Time with respect to UTC. Expected Values 0-23.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomStartMinuteOffsetEarliest

Earliest minute of day when Grooming should begin. Time with respect to UTC. Expected Values 0-1439.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomStartMinuteOffsetLatest

Latest minute of day when Grooming should begin. Time with respect to UTC. Expected Values 0-1439.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-SyncDataToCasStartHour

Hour of the day when sync to CAS via blob path should start. Default is 1 AM

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ResolutionPollTimeHours

Start time for the Resolution Poll worker.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SyncPollTimeHours

Start time for Sync Poll worker.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-DetailedSqlOutputEnabled

Determines if the SqlLog should be enabled to send SQL statements to the CDF Trace

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorQueryTimeoutSeconds

Determines the maximum time Monitoring Service will wait for the database query execution

Type:	Int32
Position:	Named
Default value:	300
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CollectHotfixDataEnabled

This setting determines if the hotfix inventory data should be collected and stored in the database or if it should be thrown away.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomApplicationInstanceRetentionDays

Determines how many days to keep the history of application instances.

Type:	Int32
Position:	Named
Default value:	0 for non-platinum, 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomApplicationErrorsRetentionDays

Determines how many days to keep the history of application errors.

Type:	Int32
Position:	Named
Default value:	1 for Platinum, Enterprise and Non-Platinum licenses
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomApplicationFaultsRetentionDays

Determines how many days to keep the history of application faults.

Type:	Int32
Position:	Named
Default value:	1 for Platinum, Enterprise and Non-Platinum licenses
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomNotificationLogRetentionDays

FIXME

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomResourceUsageRawDataRetentionDays

Determines how many days to keep Resource Utilization data.

Type:	Int32
Position:	Named
Default value:	1 for non-platinum and platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomResourceUsageMinuteDataRetentionDays

Determines how many days to keep Resource Utilization summary minute data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomResourceUsageHourDataRetentionDays

Determines how many days to keep Resource Utilization summary hour data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and 30 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomResourceUsageDayDataRetentionDays

Determines how many days to keep Resource Utilization summary day data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomProcessUsageHourDataRetentionDays

Determines how many days to keep Process Utilization summary hour data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomProcessUsageDayDataRetentionDays

Determines how many days to keep Process Utilization summary day data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and 30 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomSessionMetricsDataRetentionDays

Determines how many days to keep SessionMetrics data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomMachineMetricDataRetentionDays

Determines how many days to keep MachineMetrics data.

Type:	Int32
Position:	Named
Default value:	3 for non-platinum and platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomMachineMetricDaySummaryDataRetentionDays

Determines how many days to keep MachineMetricsSummary data.

Type:	Int32
Position:	Named
Default value:	7 for non-platinum and 90 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableDayLevelGranularityProcessUtilization

This setting is used to determine to enable or disable Day Level granularity for Process Data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomApplicationProbeLogsRetentionDays

Determines how many days to keep the history of application probe logs.

Type:	Int32
Position:	Named
Default value:	7 for Platinum, Enterprise and Non-Platinum licenses
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableHourLevelGranularityProcessUtilization

This setting is used to determine to enable or disable Hour Level granularity for Process Data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableMinLevelGranularityProcessUtilization

This setting is used to determine to enable or disable Minute Level granularity for Process Data.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomSessionAutoReconnectsRetentionDays

Determines how many days to keep SessionAutoReconnects data

Type:	Int32
Position:	Named
Default value:	7 for non-platinum, 31 for platinum.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisableHypervisorMonitoring

Disable Hypervisor Monitoring for Cloud

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SendProcessDataToCASAndDatabase

This setting is useful for sending VDA Process data to CAS and Database

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SendProcessDataToCASAndSkipDatabase

This setting is useful for sending VDA Process data to CAS and skip Database

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisablePendoTrackEvents

Disable Pendo Track Events

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableProcessDataLimitedMachines

Enable Process Data to be sent for 5000 Machines when number of VDAs is greater than 5k

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SessionRecordingDataProcessing

This setting is useful for Processing Session Recording Events that are coming from VDA

Type:	Boolean
Position:	Named
Default value:	True
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomProcessDataLimitedMachinesRetentionDays

Delete all the table entry

Type:	Int32
Position:	Named
Default value:	7 Days
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProcessDataLimitedMachinesCount

Number of Machines to be allowed for sending the Process Data

Type:	Int32
Position:	Named
Default value:	5000
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-GroomLhcReportRetentionDays

Determines how many days to keep LhcReport data

Type:	Int32
Position:	Named
Default value:	90
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-MonitorConfiguration

Set-MonitorDBConnection

March 11, 2024

Configures a database connection for the Monitor Service.

Syntax

```
Set-MonitorDBConnection
   [[-DataStore] <String>]
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Monitor Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-MonitorDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-MonitorDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Monitor Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-MonitorDBConnection cmdlet returns an object describing the status of the Monitor Service together with extra diagnostics information. Possible values are:

- OK:

The Monitor Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Monitor Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Monitor Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Monitor Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Monitor Service currently in use is newer than, and incompatible with, the version of the Monitor Service schema on the database. Upgrade the Monitor Service to a more recent version.

- DBOlderVersionThanService:

The version of the Monitor Service schema on the database is newer than, and incompatible with, the version of the Monitor Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Monitor Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Monitor Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Monitor Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

DatabaseError

An error occurred in the service while attempting a database operation.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorServiceStatus
- Get-MonitorDBConnection
- Test-MonitorDBConnection

Set-MonitorDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Monitor Service.

Syntax

```
Set-MonitorDBCredentials
    [[-DataStore] <String>]
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-MonitorDBCredentials
    [[-DataStore] <String>]
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Monitor Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Monitor Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-MonitorDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-MonitorDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-MonitorDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-MonitorDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either 'Site' or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-MonitorDBSchema
- Set-MonitorDBConnection
- Get-Credential

Set-MonitorNotificationEmailServerConfiguration

March 11, 2024

Saves a new email server configuration

Syntax

```
Set-MonitorNotificationEmailServerConfiguration
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorNotificationEmailServerConfiguration
   [-InputObject] <MonitorNotificationEmailServerConfiguration>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-MonitorNotificationEmailServerConfiguration
   -ProtocolType <EmailProtocolType>
   -ServerName <String>
   -PortNumber <Int32>
   -SenderEmailAddress <String>
   -RequiresAuthentication <Boolean>
   [-Credential <PSCredential>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The saved configuration describes the new settings to be used to send emails

Examples

EXAMPLE 1

Set new email server configration using the specified parameters

```
$secpasswd = ConvertTo-SecureString "PasswordHere" -AsPlainText -Force
$mycreds = New-Object System.Management.Automation.PSCredential ("
   username", $secpasswd)

Set-MonitorNotificationEmailServerConfiguration -ProtocolType Smtp -
   ServerName "mail.citrix.com" -PortNumber 1111 -SenderEmailAddress "
   user1@abc.com" -RequiresAuthentication $true -Credential $mycreds
```

Parameters

-InputObject

Configuration object to persist

Type:	MonitorNotificationEmailServerConfiguration
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ProtocolType

Email protocol type.

Possible values are

Smtp

SmtpSsl

SmtpTls

Type:	EmailProtocolType
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

Email server name

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PortNumber

Email server port number

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SenderEmailAddress

Sender's email address

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RequiresAuthentication

Whether the server requires authentication

Type:	Boolean
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credential

Configuration credential

Type:	PSCredential
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationEmailServerConfiguration

The saved configuration

Related Links

- Get-MonitorNotificationEmailServerConfiguration

Set-MonitorNotificationPolicy

March 11, 2024

Set/Modify MonitorNotificationPolicy object

Syntax

```
Set-MonitorNotificationPolicy
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorNotificationPolicy
   -InputObject <MonitorNotificationPolicy>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorNotificationPolicy
   -Uid <Int64>
   [-Name <String>]
   [-Description <String>]
   [-Webhook <String>]
   [-IsSnmpEnabled <Boolean>]
   [-Enabled <Boolean>]
```



```
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns a new policy instance using the specified parameters

Examples

EXAMPLE 1

Enable the policy with the id 1

```
Set-MonitorNotificationPolicy -Id 1 -Enable
```

EXAMPLE 2

Update the policy with the id 1 to add new email address

```
$policy = Get-MonitorNotificationPolicy -Uid 1

$policy.EmailAddresses += "newemail@abc.com"

Set-MonitorNotificationPolicy -InputObject $policy
```

EXAMPLE 3

Update the policy with the id 1 to add new target value

```
$policy = Get-MonitorNotificationPolicy -Uid 1
$policy.TargetIds += "766cde70-3c69-4481-a658-4e11247ac70c"

$Policy = Set-MonitorNotificationPolicy -Id 1 -InputObject $policy
```

Parameters

-InputObject

Specifies the new policy object to adjust.

Type:	MonitorNotificationPolicy
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Uid

Unique identifier for policy to bet set.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Name

New Name of the policy.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Description

String value representing the new Policy description

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Webhook

String value representing the new Policy webhook

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IsSnmpEnabled

boolean value representing if SNMP is enabled for new Policy

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Boolean paramter indicating the enabled state of the policy. true - Enabled, false - Disabled

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationPolicy

Returns the modified policy

Related Links

- Get-MonitorNotificationPolicy
- New-MonitorNotificationPolicy
- Remove-MonitorNotificationPolicy

Set-MonitorNotificationSnmpServerConfiguration

March 11, 2024

Saves a new Snmp server configuration

Syntax

```
Set-MonitorNotificationSnmpServerConfiguration
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorNotificationSnmpServerConfiguration
   [-InputObject] <MonitorNotificationSnmpServerConfiguration>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorNotificationSnmpServerConfiguration
   [-ServerName <String>]
   [-PortNumber <Int32>]
   [-SnmpSender <String>]
```



```
    [-EngineId <String>]
    [-AuthPassword <SecureString>]
    [-PrivPassword <SecureString>]
    [-PrivPasswordProtocol <PrivacyProtocolType>]
    [-AuthPasswordProtocol <AuthProtocolType>]
    [-CommunityString <String>]
    [-Protocol <SnmpProtocolType>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

The saved configuration describes the new settings to be used to send Snmps

Examples

EXAMPLE 1

Set new Snmp server configration using the specified parameters

```
Set-MonitorNotificationSnmpServerConfiguration -ServerName "
    192.168.100.100" -PortNumber 162 -SnmpSender directoradmin -
    AuthPassword authpassword -AuthPasswordProtocol MD5 -PrivatePassword
     PrivPassword -PrivatePasswordProtocol DES
```

Parameters

-InputObject

Configuration object to persist

Type:	MonitorNotificationSnmpServerConfiguration
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

Snmp listener server name

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PortNumber

Snmp server port number

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SnmpSender

Sender's Name

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EngineId

Engine ID

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AuthPassword

Password used for authentication

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrivPassword

Password used for encryption

Type:	SecureString
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PrivPasswordProtocol

Protocol used for encryption

Type:	PrivacyProtocolType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AuthPasswordProtocol

Protocol used for hash Authentication

Type:	AuthProtocolType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CommunityString

Comunity String for SNMP V2

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Protocol

Snmp protocol used for trap

Type:	SnmpProtocolType
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

MonitorNotificationSnmpServerConfiguration

The saved configuration

Related Links

- Get-MonitorNotificationSnmpServerConfiguration

Set-MonitorServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-MonitorServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-MonitorServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
```



```
    -Value <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-MonitorServiceMetadata
    [-InputObject] <Service[]>
    -Map <PSObject>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-MonitorServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                          Value
---                                          -----
property                                     value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)

Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-MonitorServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Remove-MonitorServiceMetadata

Test-MonitorDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Monitor Service.

Syntax

```
Test-MonitorDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-MonitorDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-MonitorDBConnection
    [[-DataStore] <String>]
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Monitor Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Monitor SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-MonitorDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Monitor Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

Specifies the logical name of the data store for the Monitor Service. Can be either be 'Site' or the logical name of the secondary data store.

Type:	String
Position:	3
Default value:	Site
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-MonitorDBConnection cmdlet returns an object describing the status of the selected Monitor Service instance that would result if the connection string were used with the Set-MonitorDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-MonitorDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Monitor Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Monitor Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Monitor Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Monitor Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Monitor Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Monitor Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseError

An error occurred in the service while attempting a database operation.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_MonitorMonitorSnapin
- Get-MonitorServiceStatus
- Get-MonitorDBConnection
- Set-MonitorDBConnection

Test-MonitorNotificationEmailServerConfiguration

March 11, 2024

FIXME

Syntax

```
Test-MonitorNotificationEmailServerConfiguration
    [-InputObject] <MonitorNotificationEmailServerConfiguration>
    [-Credential <PSCredential>]
    -EmailAddresses <String[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-MonitorNotificationEmailServerConfiguration
    -ProtocolType <EmailProtocolType>
    -ServerName <String>
    -PortNumber <Int32>
    -SenderEmailAddress <String>
    -RequiresAuthentication <Boolean>
    [-Credential <PSCredential>]
```



```
    -EmailAddresses <String[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-MonitorNotificationEmailServerConfiguration
    -EmailAddresses <String[]>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

FIXME

Examples

EXAMPLE 1

Test whether email can be sent using the specified oncfiguration

```
$secpasswd = ConvertTo-SecureString "PasswordHere" -AsPlainText -Force
$mycreds = New-Object System.Management.Automation.PSCredential ("
    username", $secpasswd)

Test-MonitorNotificationEmailServerConfiguration -ProtocolType Smtp -
    ServerName "mail.abc.com" -PortNumber 25 -SenderEmailAddress "
    user1@citrix.com" -RequiresAuthentication $true -EmailAddresses "
    user2@abc.com" -Credential $mycreds
```

Parameters

-InputObject

Configuration object to test

Type:	MonitorNotificationEmailServerConfiguration
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-EmailAddresses

Email addresses to which the test mail to be sent

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ProtocolType

Email protocol Possible values are Smtp SmtpSsl SmtpTls

Type:	EmailProtocolType
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

Email server name

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-PortNumber

Email server port number

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SenderEmailAddress

Sender's email address

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-RequiresAuthentication

Whether the server requires authentication

Type:	Boolean
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credential

Configuration credential

Type:	PSCredential
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

True - when test mail is successfully sent, otherwise false

Related Links

- Get-MonitorNotificationEmailServerConfiguration
- Set-MonitorNotificationEmailServerConfiguration

about_OrchOrchestrationSnapIn

March 11, 2024

Topic

about_OrchOrchestrationSnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'Orch'.

Long Description

This service long description

about_OrchOrchestrationSnapIn

March 11, 2024

Topic

about_OrchOrchestrationSnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'Orch'.

Long Description

This service long description

about_Orch_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy 'Name,-Count'

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                  # Matches Abc*
Get-<Noun> -Company "Abc`*"                   # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```


Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for

objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings

formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Get-OrchBackupRestoreValue

March 11, 2024

Sets a backup restore value to int, string or both.

Syntax

```
Get-OrchBackupRestoreValue
   -Name <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets a backup restore value.

Examples

EXAMPLE 1

Gets the Backup Max Pinned Backups.

```
Get-OrchBackupRestoreValue -Name MaxPinnedBackups
```

Parameters

-Name

Specifies the name of the backup/restore to get.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-BackupRestoreValue cmdlet returns an object containing queried name's value.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchBackupRestoreValue
- Get-OrchBackupRestoreValues

Get-OrchBackupRestoreValues

March 11, 2024

Gets all backup restore values.

Syntax

```
Get-OrchBackupRestoreValues
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets all backup restore values.

Examples

EXAMPLE 1

Gets all of the the backup/Restore names and their values.

```
Get-OrchBackupRestoreValues
```

Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-BackupRestoreValue cmdlet returns an object containing queried name's value.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchBackupRestoreValue
- Get-OrchBackupRestoreValue

Get-OrchDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Orchestration Service.

Syntax

```
Get-OrchDBConnection
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected Orchestration Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Orchestration Service instance running on controller "controller1.mydomain.net".

```
Get-OrchDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Orchestration Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the OrchestrationService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchDBConnection
- Get-OrchServiceStatus
- Test-OrchDBConnection

Get-OrchDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Orchestration Service.

Syntax

```
Get-OrchDBSchema
   [-DatabaseName <String>]
   [-ServiceGroupName <String>]
   [-ScriptType <ScriptTypes>]
   [-LocalDatabase]
   [-Sid <String>]
   [-DatabaseRights <String>]
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets SQL scripts that can be used to create a new Citrix Orchestration Service database schema, add a new Orchestration service to an existing site, remove a Orchestration service from a site, or create a database server logon for a Orchestration service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Orchestration service instance, otherwise the scripts relate to Orchestration service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Orchestration service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Orchestration service roles

If ScriptType is Evict, the returned script contains:

- Removal of Orchestration service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Orchestration Service and copies it to a file called "C:\OrchestrationSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Orchestration service schema.

```
Get-OrchDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\OrchestrationSchema.sql
```

EXAMPLE 2

Gets a script to create the appropriate database server logon for the Orchestration service. This can be used when configuring a mirror server for use.

```
Get-OrchDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
    OrchestrationLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Orchestration service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Orchestration services that share the same database instance and are

considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Orchestration Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Orchestration Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Orchestration Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Orchestration Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the required permissions for local services to access the database schema for Orchestration services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Orchestration Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)

Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchDBConnection
- Test-OrchDBConnection

Get-OrchDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Orchestration Service.

Syntax

```
Get-OrchDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Orchestration Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Orchestration Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Orchestration Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Orchestration Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Orchestration Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-OrchServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-OrchDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Orchestration Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-OrchDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Orchestration Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shut down during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Orchestration services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-OrchServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchInstalledDBVersion
- Get-OrchServiceStatus
- Get-OrchDBSchema

Get-OrchInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Orchestration Service.

Syntax

```
Get-OrchInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Orchestration Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-OrchDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Orchestration Service database schema version number.

```
Get-OrchInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Orchestration Service database schema for which upgrade scripts are supplied.

```
Get-OrchInstalledDBVersion -Upgrade
```

Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-OrchInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Orchestration

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchDBVersionChangeScript
- Get-OrchDBSchema

Get-OrchService

March 11, 2024

Gets the service record entries for the Orchestration Service.

Syntax

```
Get-OrchService
   [-Metadata <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns instances of the Orchestration Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Orchestration Service running in the current service group.

```
Get-OrchService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
```



```
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Orch_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Orch_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Orchestration.Sdk.Service

The Get-OrchServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin

Get-OrchServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Orchestration Service on the controller.

Syntax

```
Get-OrchServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Orchestration Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Orchestration Service.

```
Get-OrchServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin

Get-OrchServiceInstance

March 11, 2024

Gets the service instance entries for the Orchestration Service.

Syntax

```
Get-OrchServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Orchestration Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Orchestration Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-OrchServiceInstance
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Orchestration.Sdk.ServiceInstance

The Get-OrchServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Orch.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Orchestration.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchServiceStatus
- Reset-OrchServiceGroupMembership

Get-OrchServiceStatus

March 11, 2024

Gets the current state of the Orchestration Service on the controller.

Syntax

```
Get-OrchServiceStatus
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Enables the status of the Orchestration Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Orchestration Service.

```
1  Get-OrchServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-OrchServiceStatus command returns an object containing the status of the Orchestration Service together with extra diagnostics information.

DBUnconfigured

The Orchestration Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Orchestration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Orchestration Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Orchestration is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Orchestration is connected to a database that is valid, but it does not have the full functionality required so the Orchestration cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Orchestration Service currently in use is incompatible with the version of the Orchestration Service schema on the database. Upgrade the Orchestration Service to a more recent version.

DBOlderVersionThanService

The version of the Orchestration Service schema on the database is incompatible with the version of the Orchestration Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Orchestration Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Orchestration Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Orchestration and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Orchestration service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchDBConnection
- Test-OrchDBConnection
- Get-OrchDBConnection
- Get-OrchDBSchema

Remove-OrchServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-OrchServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-OrchServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-OrchServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-OrchServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```


Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.

```
Get-OrchService | % {
  Remove-OrchServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Set-OrchServiceMetadata

Reset-OrchEnabledFeatureList

March 11, 2024

Refreshes the Orchestration service's list of enabled features.

Syntax

```
Reset-OrchEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Orchestration Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. However, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Orchestration service instance's list of enabled features.

```
Reset-OrchEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-OrchServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Orchestration Service.

Syntax

```
Reset-OrchServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Orchestration Service access permissions and configuration service locations. The Reset-OrchServiceGroupMembership command must be run on at least one instance of the service type (Orch) after installation and registration with the configuration service. Without this operation, the Orchestration services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-OrchServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    OrchServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-OrchServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Orchestration.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-OrchServiceGroupMembership command.

Outputs

Citrix.Orchestration.Sdk.ServiceInstance

Reset-OrchServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Orchestration Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchServiceInstance
- Get-OrchServiceStatus

Set-OrchBackupRestoreValue

March 11, 2024

Sets a backup restore value to int, string or both.

Syntax

```
Set-OrchBackupRestoreValue
   -Name <String>
    [-StringValue <String>]
    [-IntValue <Int32>]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Sets a backup restore value to int, string or both.

Examples

EXAMPLE 1

Sets the Backup Max Pinned Backups value to 10.

```
Set-OrchBackupRestoreValue -Name MaxPinnedBackups 10
```

Parameters

-Name

Specifies the name of the backup/restore to set.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StringValue

The string value to set. Either a string or int value must be specified.

Type:	String
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IntValue

The int value to set. Either a string or int value must be specified.

Type:	Int32
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-BackupRestoreValue cmdlet returns an object describing the success or failure of the set call.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchBackupRestoreValue
- Get-OrchBackupRestoreValues

Set-OrchDBConnection

March 11, 2024

Configures a database connection for the Orchestration Service.

Syntax

```
Set-OrchDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Orchestration Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-OrchDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-OrchDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Orchestration Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter

Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-OrchDBConnection cmdlet returns an object describing the status of the Orchestration Service together with extra diagnostics information. Possible values are:

- OK:

The Orchestration Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Orchestration Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Orchestration Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Orchestration Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Orchestration Service currently in use is newer than, and incompatible with, the version of the Orchestration Service schema on the database. Upgrade the Orchestration Service to a more recent version.

- DBOlderVersionThanService:

The version of the Orchestration Service schema on the database is newer than, and incompatible with, the version of the Orchestration Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Orchestration Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Orchestration Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Orchestration Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchServiceStatus
- Get-OrchDBConnection
- Test-OrchDBConnection

Set-OrchDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Orchestration Service.

Syntax

```
Set-OrchDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-OrchDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Specifies SQL credentials to be used by the currently selected Citrix Orchestration Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Orchestration Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-OrchDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-OrchDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-OrchDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.


```
Set-OrchDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-OrchDBSchema
- Set-OrchDBConnection
- Get-Credential

Set-OrchServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-OrchServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-OrchServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-OrchServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-OrchServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-OrchServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                          Value
---                                          -----
property                                     value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-OrchServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Remove-OrchServiceMetadata

Start-OrchRestApi

March 11, 2024

Starts the Orchestration Service REST API.

Syntax

```
Start-OrchRestApi
     [-LoggingId <Guid>]
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```


Description

Initializes the PowerShell business logic and starts the REST API that is hosted by the selected instance of the Orchestration Service.

Examples

EXAMPLE 1

Starts the Orchestration Service REST API.

```
Start-OrchRestApi
```

Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned. Error Codes

---

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchService
- Stop-OrchRestApi

Stop-OrchRestApi

March 11, 2024

Stops the Orchestration Service REST API.

Syntax

```
Stop-OrchRestApi
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Disposes the PowerShell business logic and stops the REST API that is hosted by the selected instance of the Orchestration Service.

Examples

EXAMPLE 1

Stops the Orchestration Service REST API.

```
Stop-OrchRestApi
```


Parameters

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned. Error Codes

---

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchService
- Start-OrchRestApi

Test-OrchDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Orchestration Service.

Syntax

```
Test-OrchDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-OrchDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-OrchDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Orchestration Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Orchestration SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.


```
Test-OrchDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Orchestration Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-OrchDBConnection cmdlet returns an object describing the status of the selected Orchestration Service instance that would result if the connection string were used with the Set-OrchDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-OrchDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Orchestration Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Orchestration Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Orchestration Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Orchestration Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Orchestration Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Orchestration Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_OrchOrchestrationSnapin
- Get-OrchServiceStatus
- Get-OrchDBConnection
- Set-OrchDBConnection

about_SfStorefrontSnapIn

March 11, 2024

Topic

about_SfStorefrontSnapIn

Short Description

This Service Short Description

Command Prefix

All commands in this snap-in have the noun prefixed with ‘Sf’.

Long Description

This service long description

about_SfStorefrontSnapIn

March 11, 2024

Topic

about_SfStorefrontSnapIn

Short Description

This Service Short Description

Command Prefix

All commands in this snap-in have the noun prefixed with ‘Sf’.

Long Description

This service long description

about_Sf_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying

the name with an ordered list of elements or their numeric values,
or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy 'MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id'

-Filter <String>

This parameter lets you specify advanced filter expressions, and
supports combination of conditions with -and and -or, and grouping
with braces. For example:

Get-<Noun> -Filter 'Name -like ''High*''-or (Priority -eq 1 -and Severity -ge 2)'

The syntax is close enough to PowerShell syntax that you can use
script blocks in most cases. This can be easier to read as it reduces
quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double
quotes, and omit the quotes altogether for many strings. The order of
parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company ''citrix''-Product '[X]EN*'
Get-<Noun> -Product ''Xen*''-Company ''CITRIX''
Get-<Noun> -Filter { Company -eq 'Citrix'-and Product -like 'Xen*'}

See about_Quoting_Rules and about_Wildcards for details about PowerShell
handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape
the wildcards using the normal PowerShell escape mechanisms (see
about_Escape_Characters), or switch to a filter expression and the -eq
operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
```



```
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter 'Capacity -ge 10gb'
Get-<Noun> -Filter 'Age -ge 20 -and Age -lt 40'
Get-<Noun> -Filter 'VolumeLevel -like "[123]"'
Get-<Noun> -Filter 'Enabled -ne $false'
Get-<Noun> -Filter 'NullableProperty -ne $null'

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter 'Enabled'# Equivalent to 'Enabled -eq $true'
Get-<Noun> -Filter '-not Enabled'# Equivalent to 'Enabled -eq $false'

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq "Circle"}
Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan

values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```


Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid


```
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, "-1:30"means 1 hour and 30 minutes ago.

Add-SfServerToCluster

March 11, 2024

Adds a new server to an existing cluster.

Syntax

```
Add-SfServerToCluster
   [-StorefrontUrl <Uri>]
   -ClusterId <Guid>
   -ServerName <String>
   [-FarmName <String>]
   [-XmlServices <Uri[]>]
   [-RunAsynchronously <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Adds a new server to an existing cluster. Optionally updates Farm and Storefront Url. After operation succeeds, all servers are configured identically.

Examples

EXAMPLE 1

Adds "NewSfServer"to cluster with id (Guid).


```
Add-SfServerToCluster -ClusterId (Guid) -ServerName NewSfServer -
    RunAsynchronously $true
```

Parameters

-ClusterId

The id of the cluster to perform operation on.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

The name of the server to join to exisiting cluster. The name must be one of the values returned by Get-SfCluster

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StorefrontUrl

The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

Type:	Uri
Position:	Named

Default value:	Server name and http binding.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FarmName

Name of the farm that will be used within Store service. Either both FarmName and XmlServices need to be specified or none of them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-XmlServices

Collection of the url of xml services that will be used inside a farm. The urls neeed to be http or https, be absolute and share the same schema and port. Either both FarmName and XmlServices need to be specified or none of them.

Type:	Uri[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

If set, the command will run asynchronously.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.Task or Citrix.Storefront.DataModel.Cluster

Returns cluster description or a task, if ran asynchronously.

Related Links

- Get-SfCluster
- New-SfCluster
- Remove-SfServerFromCluster
- Set-SfCluster

Get-SfCluster

March 11, 2024

Gets all Storefront clusters present in the site.

Syntax

```
Get-SfCluster
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets all Storefront clusters present in the site. There is one special Cluster with null id that lists the servers that are not part to any cluster (they are available to join any).

Examples

EXAMPLE 1

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.DataModel.Cluster[]

Returns array of clusters available in current deployment.

Related Links

- New-SfCluster
- Add-SfServerToCluster
- Remove-SfServerFromCluster
- Set-SfCluster

Get-SfDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Storefront Service.

Syntax

```
Get-SfDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected Storefront Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Storefront Service instance running on controller "controller1.mydomain.net".

```
Get-SfDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Storefront Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the StorefrontService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Set-SfDBConnection
- Get-SfServiceStatus
- Test-SfDBConnection

Get-SfDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Storefront Service.

Syntax

```
Get-SfDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Storefront Service database schema, add a new Storefront service to an existing site, remove a Storefront service from a site, or create a database server logon for a Storefront service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Storefront service instance, otherwise the scripts relate to Storefront service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Storefront service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Storefront service roles

If ScriptType is Evict, the returned script contains:

- Removal of Storefront service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Storefront Service and copies it to a file called "C:\StorefrontSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Storefront service schema.

```
Get-SfDBSchema -DatabaseName MySiteDB -ServiceGroupName  MyServiceGroup
    > C:\StorefrontSchema.sql
```


EXAMPLE 2

Gets a script to create the appropriate database server logon for the Storefront service. This can be used when configuring a mirror server for use.

```
1  Get-SfDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
       StorefrontLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Storefront service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Storefront services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Storefront Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Storefront Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Storefront Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Storefront Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the

required permissions for local services to access the database schema for Storefront services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Storefront Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Set-SfDBConnection
- Test-SfDBConnection

Get-SfDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Storefront Service.

Syntax

```
Get-SfDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Storefront Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Storefront Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Storefront Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Storefront Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Storefront Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-SfServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-SfDBVersionChangeScript -DatabaseName MyDb -TargetVersion
    7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Storefront Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-SfDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Storefront Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Storefront services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-SfServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfInstalledDBVersion
- Get-SfServiceStatus
- Get-SfDBSchema

Get-SfInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Storefront Service.

Syntax

```
Get-SfInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Storefront Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-SfDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Storefront Service database schema version number.

```
Get-SfInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Storefront Service database schema for which upgrade scripts are supplied.

```
Get-SfInstalledDBVersion -Upgrade
```


Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-SfInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Storefront

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfDBVersionChangeScript
- Get-SfDBSchema

Get-SfIsStorefrontInstalled

March 11, 2024

Tells whether StoreFront Services and Privileged Service are installed.

Syntax

```
Get-SfIsStorefrontInstalled
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Examples

EXAMPLE 1

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

bool

True if both StoreFront Services and Privileged Service are installed, false otherwise.

Related Links

Get-SfOptimalGateway

March 11, 2024

Gets the configured optimal gateways for farms.

Syntax

```
Get-SfOptimalGateway
   -SiteId <Int64>
   -ResourcesVirtualPath <String>
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Gets the configured optimal gateways for farms.

Examples

EXAMPLE 1

```
Get-DSOptimalGatewayForFarms -siteId 1 -ResourcesVirtualPath "/Citrix/
    MyStore"
```

Parameters

-SiteId

Site ID within IIS. This is typically 1 for the site in IIS where StoreFront is installed by default.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ResourcesVirtualPath

Path to the store that is to be configured to have a farm to optimal gateway mapping.

Example: “/Citrix/Store”

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.OptimalGateway

Related Links

- Set-SfOptimalGateway
- Remove-SfOptimalGateway

Get-SfService

March 11, 2024

Gets the service record entries for the Storefront Service.

Syntax

```
Get-SfService
    [-Metadata <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
```



```
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Returns instances of the Storefront Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Storefront Service running in the current service group.

```
Get-SfService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata "abc:x*"matches records with a metadata entry having a key name of "abc"and a value starting with the letter "x".

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Sf_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Sf_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.Service

The Get-SfServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin

Get-SfServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Storefront Service on the controller.

Syntax

```
Get-SfServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Storefront Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Storefront Service.

```
Get-SfServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin

Get-SfServiceInstance

March 11, 2024

Gets the service instance entries for the Storefront Service.

Syntax

```
Get-SfServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns service interfaces published by instances of the Storefront Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Storefront Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
Get-SfServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.ServiceInstance

The Get-SfServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Sf.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Storefront.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfServiceStatus
- Reset-SfServiceGroupMembership

Get-SfServiceStatus

March 11, 2024

Gets the current state of the Storefront Service on the controller.

Syntax

```
Get-SfServiceStatus
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables the status of the Storefront Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then

it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Storefront Service.

```
Get-SfServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-SfServiceStatus command returns an object containing the status of the Storefront Service together with extra diagnostics information.

DBUnconfigured

The Storefront Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Storefront Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Storefront Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Storefront is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Storefront is connected to a database that is valid, but it does not have the full functionality required so the Storefront cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Storefront Service currently in use is incompatible with the version of the Storefront Service schema on the database. Upgrade the Storefront Service to a more recent version.

DBOlderVersionThanService

The version of the Storefront Service schema on the database is incompatible with the version of the Storefront Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Storefront Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Storefront Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Storefront and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Storefront service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Set-SfDBConnection
- Test-SfDBConnection
- Get-SfDBConnection
- Get-SfDBSchema

Get-SfTask

March 11, 2024

Gets the task history for the Storefront Service.

Syntax

```
Get-SfTask
   [[-TaskId] <Guid>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Returns a list of tasks that have run or are currently running within the Storefront Service.

Examples

EXAMPLE 1

Parameters

-TaskId

Specifies the task identifier to be returned.

Type:	Guid
Position:	2
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	True

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Sf_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Sf_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapIn
- Remove-SfTask
- Set-SfTaskMetadata
- Remove-SfTaskMetadata

New-SfCluster

March 11, 2024

Creates new Storefront cluster with default set of services.

Syntax

```
New-SfCluster
   [-StorefrontUrl <Uri>]
   -ServerName <String>
   -FarmName <String>
   -XmlServices <Uri[]>
   [-RunAsynchronously <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

The server will have a default set of services containing fully-functionaly Storefront server with Authentication, Store, Receiver for Web and Desktop Appliance.

Examples

EXAMPLE 1

Creates a new cluster on server “SfServer”. The server will have a default set of services containing fully-functionaly Storefront server with Authentication, Store, Receiver for Web, Desktop Appliance site and the required infrastructure. The Store service will have a farm named “XdSiteName”that will contain servers farm1 and farm2, whose will be contacted using http on default port 80.


```
New-SfCluster -FarmName "XdSiteName" -XmlServices http://farm1,http://
    farm2 -ServerName SfServer -RunAsynchronously $true
```

Parameters

-ServerName

The name of the server to build a cluster on. The name must be one of the values returned by Get-SfCluster

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-FarmName

Name of the farm that will be used within Store service.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-XmlServices

Collection of the url of xml services that will be used inside a farm. The urls neeed to be http or https, be absolute and share the same schema and port.

Type:	Uri[]

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StorefrontUrl

The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

Type:	Uri
Position:	Named
Default value:	Server name and http binding.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

If set, the command will run asynchronously.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.Task or Citrix.Storefront.DataModel.Cluster

Returns cluster description or a task, if ran asynchronously.

Related Links

- Get-SfCluster

- Add-SfServerToCluster
- Remove-SfServerFromCluster
- Set-SfCluster

Remove-SfOptimalGateway

March 11, 2024

Removes the optimal gateway for farms configuration.

Syntax

```
Remove-SfOptimalGateway
       -SiteId <Int64>
       -ResourcesVirtualPath <String>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Removes the optimal gateway for farms configuration.

Examples

EXAMPLE 1

```
Remove-DSOptimalGatewayForFarms -siteId 1 -ResourcesVirtualPath "/
   Citrix/MyStore"
```

Parameters

-SiteId

Site ID within IIS. This is typically 1 for the site in IIS where StoreFront is installed by default.

Type:	Int64
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ResourcesVirtualPath

Path to the store that is to be configured to have a farm to optimal gateway mapping.

Example: "/Citrix/Store"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-SfOptimalGateway
- Set-SfOptimalGateway

Remove-SfServerFromCluster

March 11, 2024

Removes server from the cluster.

Syntax

```
Remove-SfServerFromCluster
      -ClusterId <Guid>
      [-StorefrontUrl <Uri>]
      -ServerName <String>
      [-FarmName <String>]
      [-XmlServices <Uri[]>]
      [-RunAsynchronously <Boolean>]
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Removes server from the cluster and propagates information to other servers. The configuration of the server is wiped out, so the server can be reused.

Examples

EXAMPLE 1

Removes Server "BrokenSfServer"from a Cluster with id (Guid).

```
Remove-SfServerFromCluster -ClusterId (Guid) -ServerName BrokenSfServer
    -RunAsynchronously $true
```

Parameters

-ClusterId

The if of the cluster to perform operation on.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServerName

The name of the server to remove from cluster. The name must be one of the values returned by Get-SfCluster.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StorefrontUrl

The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

Type:	Uri
Position:	Named
Default value:	Server name and http binding.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FarmName

Name of the farm that will be used within Store service. Either both FarmName and XmlServices need to be specified or none of them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-XmlServices

Collection of the url of xml services that will be used inside a farm. The urls neeed to be http or https, be absolute and share the same schema and port. Either both FarmName and XmlServices need to be specified or none of them.

Type:	Uri[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

If set, the command will run asynchronously.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.Task or Citrix.Storefront.DataModel.Cluster

Returns cluster description or a task, if ran asynchronously.

Related Links

- Get-SfCluster
- New-SfCluster
- Add-SfServerToCluster
- Set-SfCluster

Remove-SfServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-SfServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-SfServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-SfServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-SfServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.


```
Get-SfService | % {
  Remove-SfServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Set-SfServiceMetadata

Remove-SfTask

March 11, 2024

Removes from the database completed tasks for the Storefront Service.

Syntax

```
Remove-SfTask
       [-TaskId] <Guid>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```


Description

Enables completed tasks that have run within the Storefront Service to be removed from the database.

Examples

EXAMPLE 1

Parameters

-TaskId

Specifies the identifier for the task to be removed.

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

PSObject

Objects containing the TaskId parameter can be piped to the Remove-SfTask command.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

ServiceStatusInvalidDb

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration service.

OperationDeniedByConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapIn
- Get-SfTask
- Set-SfTaskMetadata
- Remove-SfTaskMetadata

Remove-SfTaskMetadata

March 11, 2024

Removes metadata from the given Task.

Syntax

```
Remove-SfTaskMetadata
      [-TaskId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
```



```
5         [<CitrixCommonParameters>]
6         [<CommonParameters>]
```

```
1 Remove-SfTaskMetadata
2       [-TaskId] <Guid>
3       -Map <PSObject>
4       [-LoggingId <Guid>]
5       [<CitrixCommonParameters>]
6       [<CommonParameters>]
```

```
1 Remove-SfTaskMetadata
2       [-InputObject] <Task[]>
3       -Name <String>
4       [-LoggingId <Guid>]
5       [<CitrixCommonParameters>]
6       [<CommonParameters>]
```

```
1 Remove-SfTaskMetadata
2       [-InputObject] <Task[]>
3       -Map <PSObject>
4       [-LoggingId <Guid>]
5       [<CitrixCommonParameters>]
6       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Task.

Examples

EXAMPLE 1

Remove all metadata from all Task objects.

```
1 Get-SfTask | % {
2   Remove-SfTaskMetadata -Map $_.MetadataMap  }
```

Parameters

-TaskId

Id of the Task

Type: Guid

Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	Task[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “Sys-

tem.Collections.Generic.Dictionary[String,String]"). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapIn
- Set-SfTaskMetadata

Reset-SfEnabledFeatureList

March 11, 2024

Refreshes the Storefront service's list of enabled features.

Syntax

```
Reset-SfEnabledFeatureList
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Storefront Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Storefront service instance's list of enabled features.

```
Reset-SfEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-SfServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Storefront Service.

Syntax

```
Reset-SfServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Storefront Service access permissions and configuration service locations. The Reset-SfServiceGroupMembership command must be run on at least one instance of the service type (Sf) after installation and registration with the configuration service. Without this operation, the Storefront services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-SfServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    SfServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-SfServiceGroupmembership
```


Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Storefront.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-SfServiceGroupMembership command.

Outputs

Citrix.Storefront.Sdk.ServiceInstance

Reset-SfServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Storefront Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfServiceInstance
- Get-SfServiceStatus

Set-SfCluster

March 11, 2024

Sets the parameters on the given cluster.

Syntax

```
Set-SfCluster
   [-StorefrontUrl <Uri>]
   -ClusterId <Guid>
   [-FarmName <String>]
   [-XmlServices <Uri[]>]
   [-RunAsynchronously <Boolean>]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Sets the parameters on the given cluster and propagate the changes to all servers within a given cluster.

Examples

EXAMPLE 1

Sets a Storefront Url in a Cluster with id (Guid). Propagates changes to all servers that are part of the cluster.

```
Set-SfCluster -StorefrontUrl http://SfUrl -ClusterId (Guid) -
    RunAsynchronously $true
```

Parameters

-ClusterId

The if of the cluster to perform operation on.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StorefrontUrl

The url that will be used by Receivers to contact Storefront. Http or https absolute urls are accepted.

Type:	Uri
Position:	Named
Default value:	Server name and http binding.
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FarmName

Name of the farm that will be used within Store service. Either both FarmName and XmlServices need to be specified or none of them.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-XmlServices

Collection of the url of xml services that will be used inside a farm. The urls neeed to be http or https, be absolute and share the same schema and port. Either both FarmName and XmlServices need to be specified or none of them.

Type:	Uri[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RunAsynchronously

If set, the command will run asynchronously.

Type:	Boolean
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Storefront.Sdk.Task or Citrix.Storefront.DataModel.Cluster

Returns cluster description or a task, if ran asynchronously.

Related Links

- Get-SfCluster
- New-SfCluster
- Add-SfServerToCluster
- Remove-SfServerFromCluster

Set-SfDBConnection

March 11, 2024

Configures a database connection for the Storefront Service.

Syntax

```
Set-SfDBConnection
    [-DBConnection] <String>
    [-Force]
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Storefront Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-SfDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-SfDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Storefront Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-SfDBConnection cmdlet returns an object describing the status of the Storefront Service together with extra diagnostics information. Possible values are:

- OK:

The Storefront Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Storefront Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Storefront Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Storefront Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Storefront Service currently in use is newer than, and incompatible with, the version of the Storefront Service schema on the database. Upgrade the Storefront Service to a more recent version.

- DBOlderVersionThanService:

The version of the Storefront Service schema on the database is newer than, and incompatible with, the version of the Storefront Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Storefront Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Storefront Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Storefront Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfServiceStatus
- Get-SfDBConnection
- Test-SfDBConnection

Set-SfDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Storefront Service.

Syntax

```
Set-SfDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-SfDBCredentials
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Storefront Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Storefront Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-SfDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-SfDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.


```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-SfDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-SfDBCredentials $null
```

Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-SfDBSchema
- Set-SfDBConnection
- Get-Credential

Set-SfOptimalGateway

March 11, 2024

Set the farms and the optimal gateway to use for launch.

Syntax

```
Set-SfOptimalGateway
   -SiteId <Int64>
   -ResourcesVirtualPath <String>
   -HostNames <String[]>
   -StaUrls <String[]>
   -Farm <String>
   [-StasBypassDuration <TimeSpan>]
   [-StasUseLoadBalancing]
   [-EnableSessionReliability]
   [-UseTwoTickets]
   [-EnabledOnDirectAccess]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Set the farms and the optimal gateway to use for launch.

Examples

EXAMPLE 1

```
Set-DSOptimalGatewayForFarms -SiteId 1 -ResourcesVirtualPath /Citrix/
    Store -Hostnames @("gateway1.citrix.com:2222") -StaUrls @("https://
    server1.citrix.com/staurl") -StasBypassDuration "00.02:00:00" -
    EnabledOnDirectAccess
```

Parameters

-SiteId

Site ID within IIS. This is typically 1 for the site in IIS where StoreFront is installed by default.

Type:	Int64
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-ResourcesVirtualPath

Path to the store that is to be configured to have a farm to optimal gateway mapping.

Example: “/Citrix/Store”

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	True

-HostNames

Specifies the fully qualified domain name (FQDN) and port of the optimal NetScaler Gateway appliance.

Example1 for standard vServer port 443: gateway.example.com

Example2 for nonstandard vServer port 500: gateway.example.com:500

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StaUrls

Specifies the URLs for XenDesktop, XenApp, and VDI-in-a-Box servers running the Secure Ticket Authority (STA). If using multiple farms, list the STA servers on each using a comma separated list:

Example: "http://xenapp-a.ptd.com/scripts/ctxsta.dll","http://xendesktop-a.ptd.com/scripts/ctxsta.dll"

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Farm

Specificies the set of delivery controllers, as named when they were configured with StoreFront.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-StasBypassDuration

Set the time period, in hours, minutes, and seconds, for which an STA is considered unavailable after a failed request.

Type:	TimeSpan
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-StasUseLoadBalancing

Set to true: randomly obtains session tickets from all STAs, evenly distributing requests across all the STAs.

Set to false: users are connected to the first available STA in the order in which they are listed in the configuration, minimizing the number of STAs in use at any given time.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnableSessionReliability

Set to true: keeps disconnected sessions open while Receiver attempts to reconnect automatically. If you configured multiple STAs and want to ensure that session reliability is always available, set the value of the UseTwoTickets attribute to true to obtain session tickets from two different STAs in case one STA becomes unavailable during the session.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UseTwoTickets

Set to true: obtains session tickets from two different STAs in case one STA becomes unavailable during the session.

Set to false: uses only a single STA server.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-EnabledOnDirectAccess

Set to true: ensures that when local users on the internal network log on to StoreFront directly, connections to their resources are still routed through the optimal appliance defined for the farm.

Set to false: connections to resources are not routed through the optimal appliance for the farm unless users access StoreFront through a NetScaler Gateway.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-SfOptimalGateway
- Remove-SfOptimalGateway

Set-SfServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-SfServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-SfServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-SfServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-SfServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the Service with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-SfServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-SfServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Remove-SfServiceMetadata

Set-SfTaskMetadata

March 11, 2024

Adds or updates metadata on the given Task.

Syntax

```
Set-SfTaskMetadata
   [-TaskId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-SfTaskMetadata
   [-TaskId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-SfTaskMetadata
   [-InputObject] <Task[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-SfTaskMetadata
   [-InputObject] <Task[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given Task objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property' and a value of 'value' to the Task with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-SfTaskMetadata -TaskId 4CECC26E-48E1-423F-A1F0-2A06DDD0805C -Name
   property -Value value

Key                                     Value
```



```
4 ---                                     -----
5 property                                value
```

Parameters

-TaskId

Id of the Task

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	Task[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Task specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series

of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-SfTaskMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapIn
- Remove-SfTaskMetadata

Test-SfDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Storefront Service.

Syntax

```
Test-SfDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-SfDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-SfDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Storefront Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Storefront SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-SfDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Storefront Service instance.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-SfDBConnection cmdlet returns an object describing the status of the selected Storefront Service instance that would result if the connection string were used with the Set-SfDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-SfDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Storefront Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Storefront Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Storefront Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Storefront Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Storefront Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Storefront Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_SfStorefrontSnapin
- Get-SfServiceStatus
- Get-SfDBConnection
- Set-SfDBConnection

about_TrustTrustSnapIn

March 11, 2024

Topic

about_TrustTrustSnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'Trust'.

Long Description

This service long description

about_TrustTrustSnapIn

March 11, 2024

Topic

about_TrustTrustSnapin

Short Description

This service short description

Command Prefix

All commands in this snap-in have the noun prefixed with 'Trust'.

Long Description

This service long description

about_Trust_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:


```
# Escape examples
Get-<Noun> -Company "Abc[*]"                    # Matches Abc*
Get-<Noun> -Company "Abc`*"                     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}

Get-<Noun> -Filter { Shape -like ‘C*’}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
```



```
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{


```
break;
}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Confirm-TrustVdaEnrollmentToken

March 11, 2024

This cmdlet will validate or confirm that the provided VDA enrollment token is valid.

Syntax

```
Confirm-TrustVdaEnrollmentToken
        [-Token] <String>
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```

Description

If the token is valid it will assume the token is being used and therefore it will increment the total number of times the token was used to add a machine by one and update the last used date.

Examples

EXAMPLE 1

Determines if the provided VDA enrollment JWT token is valid.

```
Confirm-TrustVdaEnrollmentToken -Token {
 your vda token here }
```


Parameters

-Token

The JWT token that delegates the ability to enroll VDA machines with a particular machine catalog.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Boolean

Returns true if the token is valid or false if the token is not valid.

Related Links

Get-TrustDBConnection

March 11, 2024

Gets the database connection string for the specified data store used by the Trust Service.

Syntax

```
Get-TrustDBConnection
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the database connection string from the currently selected Trust Service instance.

If the returned string is blank, no valid connection string has been specified. In this case the service is running, but is idle and awaiting specification of a valid connection string.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

Examples

EXAMPLE 1

Gets the database connection string in use by the Trust Service instance running on controller "controller1.mydomain.net".

```
Get-TrustDBConnection -AdminAddress controller1.mydomain.net
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

The database connection string configured for the current Trust Service instance.

Notes

If the command fails, the following errors can be returned:

- NoDBConnections

  The database connection string for the TrustService has not been specified.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Set-TrustDBConnection
- Get-TrustServiceStatus
- Test-TrustDBConnection

Get-TrustDBSchema

March 11, 2024

Gets SQL scripts to create or maintain the database schema for the Citrix Trust Service.

Syntax

```
Get-TrustDBSchema
    [-DatabaseName <String>]
    [-ServiceGroupName <String>]
    [-ScriptType <ScriptTypes>]
    [-LocalDatabase]
    [-Sid <String>]
    [-DatabaseRights <String>]
    [-AzureDatabase]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets SQL scripts that can be used to create a new Citrix Trust Service database schema, add a new Trust service to an existing site, remove a Trust service from a site, or create a database server logon for a Trust service.

If no Sid parameter is provided, the scripts obtained relate to the currently selected Trust service instance, otherwise the scripts relate to Trust service instance running on the machine identified by the Sid provided. When obtaining the Evict script, a Sid parameter must be supplied.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

The service instance used to obtain the scripts does not need to be a member of a site or to have had its database connection configured.

The database scripts support only Microsoft SQL Server, or SQL Server Express, and require Windows integrated authentication to be used. They can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The ScriptType parameter determines which script is obtained. If ScriptType is not specified, or is FullDatabase, the script contains:

- Creation of service schema
- Creation of database server logon
- Creation of database user
- Addition of database user to Trust service roles

If ScriptType is Instance, the returned script contains:

- Creation of database server logon
- Creation of database user
- Addition of database user to Trust service roles

If ScriptType is Evict, the returned script contains:

- Removal of Trust service instance from database
- Removal of database user

If ScriptType is Login, the returned script contains:

- Creation of database server logon only

ScriptType Database is deprecated, and FullDatabase should be used instead.

If the service uses two data stores they can exist in the same database.

You do not need to configure a database before using this command.

Examples

EXAMPLE 1

Gets a script to create the full database schema for the Citrix Trust Service and copies it to a file called "C:\TrustSchema.sql"

This script can be used to create the service schema in a database with name "MySiteDB", which must already exist, and must not already contain a Trust service schema.

```
Get-TrustDBSchema -DatabaseName MySiteDB -ServiceGroupName
    MyServiceGroup > C:\TrustSchema.sql
```


EXAMPLE 2

Gets a script to create the appropriate database server logon for the Trust service. This can be used when configuring a mirror server for use.

```
1  Get-TrustDBSchema -DatabaseName MySiteDB -ScriptType Login > C:\
       TrustLogins.sql
```

Parameters

-DatabaseName

Specifies the name of the database into which the new Trust service schema is to be placed, or in which it already exists. The database itself is not created by any of the script types; it must already exist before the scripts are run.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceGroupName

Specifies the name of the service group to be used when creating the database schema. The service group is a collection of all the Trust services that share the same database instance and are considered equivalent; that is, all the services within a service group can be used interchangeably.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ScriptType

Specifies the type of database script returned. Available script types are:

- FullDatabase

Creates a database schema for the Citrix Trust Service in a database instance that does not already contain one. This is used when creating a new site. DatabaseName and ServiceGroupName are required parameters for this script type.

- Instance

Adds a Trust Service instance to a database and so to the associated site. Appropriate database server logons and users are created to allow the service instance access to the required service schemas.

- Evict

Removes a Trust Service instance from the database and so from the site. All reference to the service instance is removed from the database. DatabaseName and Sid are required parameters for this script type.

- Login

Adds a logon for the Trust Service instance to a database server. This is specifically for use when configuring SQL Server mirroring where the mirror server must have appropriate logons created for all service instances in the site.

- Database

This is deprecated. FullDatabase should be used instead.

Type:	ScriptTypes
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LocalDatabase

Specifies whether the database script is to be used in a database instance run on the same controller as other services in the service group. Including this parameter ensures the script creates only the

required permissions for local services to access the database schema for Trust services. If this parameter is specified inappropriately, the service instance will not be able to connect to the database.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Sid

Specifies the SID of the controller on which the Trust Service instance to remove from the database is running (only valid for a script type of Evict).

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-DatabaseRights

Specifies the right the database script should expect to be run under. Available rights are:

- Mixed

  Creates a database schema which uses all rights.

- SysAdmin

  Creates a database schema which does the minimum with the SysAdmin (sa) rights.

- DbOwner

  Creates a database schema which only needs Database Owner (dbo) rights.

  This script expects to be used after the SysAdmin script has been run.

Type:	String
Position:	Named
Default value:	Mixed
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Specifies that the generated schema must be compatible with Azure SQL limits, including not generating code for logins.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the required SQL script for applying to a database.

Notes

The scripts returned support Microsoft SQL Server Express Edition, Microsoft SQL Server Standard Edition, and Microsoft SQL Server Enterprise Edition databases only, and are generated on the assumption that integrated authentication will be used.

If the ScriptType parameter is not included or set to 'FullDatabase', the full database script is returned, which will:

- Create the database schema.
- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist).

If the ScriptType parameter is set to 'Instance', the script will:

- Create the user and the role (providing the schema does not already exist).
- Create the logon (providing the schema does not already exist) and associate it with a user.

If the ScriptType parameter is set to 'Login', the script will:

- Create the logon (providing the schema does not already exist) and associate it with a pre-existing user of the same name.

ScriptType value of 'Database'is deprecated; 'FullDatabase'should be used instead.

If the LocalDatabase parameter is included, the NetworkService account will be added to the list of accounts permitted to access the database. This is required only if the database is run on a controller.

If the command fails, the following errors can be returned:

- GetSchemasFailed

  The database schema could not be found.

- ActiveDirectoryAccountResolutionFailed

  The specified Active Directory account or Group could not be found.

- DatabaseError

  An error occurred in the service while attempting a database

- operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the

- service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Set-TrustDBConnection
- Test-TrustDBConnection

Get-TrustDBVersionChangeScript

March 11, 2024

Gets an SQL service schema update script for the Citrix Trust Service.

Syntax

```
Get-TrustDBVersionChangeScript
   -DatabaseName <String>
   -TargetVersion <Version>
   [-AzureDatabase]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Gets an SQL script that can be used to update the current Citrix Trust Service database schema. An update can be an upgrade or downgrade.

A script can be obtained to update the current service schema to any version that is reachable by applying available schema update packages that have been uploaded by the Citrix Trust Service.

Typically, this mechanism is used to update the current service schema to a newer version, however it can also be used to revert previously applied updates.

The SQL script obtained can be run using SQL Server's SQLCMD utility, or by copying the script into an SQL Server Management Studio (SSMS) query window and executing the query. If using SSMS, the query must be executed in 'SQLCMD mode'.

The schema update packages used to generate update scripts are stored in the database; because of this, any Citrix Trust Service in the site can be used to obtain a schema update script.

The fact that an update package is available in the database does not mean that the update has actually been applied to the service's schema. In addition, application of an update does not remove the associated update packages.

Take care when using the update scripts. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK. The database should be backed-up before an update is attempted. The database script may also require exclusive use of the schema, in which case all Citrix Trust Services must be shutdown before applying the update.

Once an update has been applied to the service schema, any existing Citrix Trust Services that are incompatible with the updated schema will cease to operate. The service state, as reported by Get-TrustServiceStatus, provides information about the service compatibility (e.g. DBNewerVersionThanService).

Examples

EXAMPLE 1

Gets an SQL update script to update the current schema to version 7.40.0.0. The resulting update_740.sql script is suitable for direct use with the SQL Server SQLCMD utility.

```
$update = Get-TrustDBVersionChangeScript -DatabaseName MyDb -
    TargetVersion 7.40.0.0
$update.Script > update_740.sql
```

Parameters

-DatabaseName

The name of the database containing the Citrix Trust Service schema to be updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-TargetVersion

The required target service schema version of the update. This is the service schema version obtained after the update script is applied.

Type:	Version
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AzureDatabase

Indicates that script is to update an Azure database. If this switch is not used when an Azure database is to be updated, the update will fail when an attempt is made to apply it.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

PSObject

The Get-TrustDBVersionChangeScript cmdlet returns a PSObject containing a script that can be used to update the Citrix Trust Service database schema. The object has the following properties:

- Script

  The raw text of the SQL script to apply the update.

- CanUndo

  If true, indicates that after the update has been applied, a script to revert from the updated schema to the schema state prior to the update can be obtained.

  Because Get-<#= CmdletPrefix #>DBVersionChangeScript gets only update scripts relating to the current schema version, a script to revert an update can be obtained only after the update has been applied.

- NeedExclusiveAccess

  If true, indicates that the update requires exclusive access to the Citrix <#= ServiceName #> Service's schema while the update is applied; all Citrix <#= ServiceName #> Services must be shutdown during the update.

Notes

The PSObject returned by this cmdlet contains the following properties:

- Script

  The raw text of the SQL script to apply the update, or null in the case when no upgrade path to the specified target version exists.

- NeedExclusiveAccess

  Indicates whether all services in the service group must be shut down during the update or not.

- CanUndo

  Indicates whether the generated script allows the updated schema to be reverted to the state prior to the update.

Scripts to update the schema version are stored in the database so any service in the service group can obtain these scripts. Extreme caution should be exercised when using update scripts. Citrix recommends backing up the database before attempting to upgrade the schema. Database update scripts may require exclusive use of the schema and so may not be able to execute while any Trust services are running. However, this depends on the specific update being carried out.

After a schema update has been carried out, services that require the previous version of the schema may cease to operate. The ServiceState parameter reported by the Get-TrustServiceStatus command provides information about service compatibility. For example, if the schema has been upgraded to a more recent version that a service cannot use, the service reports "DBNewerVersionThanService".

If the command fails, the following errors can be returned:

- NoOp

  The operation was successful but had no effect.

- NoDBConnections

  The database connection string for the <#= ServiceName #> Service has not been specified.

- DatabaseError

  An error occurred in the service while attempting a database operation.

- DatabaseNotConfigured

  The operation could not be completed because the database for the service is not configured.

- DataStoreException

  An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

- PermissionDenied

  You do not have permission to execute this command.

- AuthorizationError

  There was a problem communicating with the Citrix Delegated Administration Service.

- CommunicationError

  There was a problem communicating with the remote service.

- ExceptionThrown

  An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustInstalledDBVersion
- Get-TrustServiceStatus
- Get-TrustDBSchema

Get-TrustInstalledDBVersion

March 11, 2024

Gets a list of all available database schema versions for the Trust Service.

Syntax

```
Get-TrustInstalledDBVersion
    [-Upgrade]
    [-Downgrade]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Gets the current version number of the Citrix Trust Service database schema when called with no parameters.

When called with the -Upgrade parameter, gets the service schema version numbers to which an upgrade could be performed.

When called with the -Downgrade parameter, gets the service schema version numbers to which a downgrade could be performed.

The SQL scripts to perform schema upgrades and downgrades can be obtained using the Get-TrustDBVersionChangeScript cmdlet. Citrix recommends that where possible service schema upgrades are performed using Studio rather than manually via the SDK.

Only one of the -Upgrade or -Downgrade parameters may be supplied at once.

Examples

EXAMPLE 1

Gets the current Citrix Trust Service database schema version number.

```
Get-TrustInstalledDBVersion
```

EXAMPLE 2

Get the versions of the Trust Service database schema for which upgrade scripts are supplied.

```
Get-TrustInstalledDBVersion -Upgrade
```


Parameters

-Upgrade

Specifies that only schema versions to which the current database version can be updated should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Downgrade

Specifies that only schema versions to which the current database version can be reverted should be returned.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Version

Get-TrustInstalledDBVersion returns database schema version numbers as requested.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

Both the Upgrade and Downgrade flags were specified.

NoOp

The operation was successful but had no effect.

NoDBConnections

The database connection string for the Trust

Service has not been specified.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustDBVersionChangeScript
- Get-TrustDBSchema

Get-TrustService

March 11, 2024

Gets the service record entries for the Trust Service.

Syntax

```
Get-TrustService
   [-Metadata <String>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns instances of the Trust Service that the service publishes. The service records contain account security identifier information that can be used to remove each service from the database.

A database connection for the service is required to use this command.

Examples

EXAMPLE 1

Get all the instances of the Trust Service running in the current service group.

```
Get-TrustService

Uid                : 1
ServiceHostId      : aef6f464-f1ee-4042-a523-66982e0cecd0
DNSName            : MyServer.company.com
MachineName        : MYSERVER
CurrentState       : On
LastStartTime      : 04/04/2011 15:25:38
LastActivityTime   : 04/04/2011 15:33:39
OSType             : Win32NT
OSVersion          : 6.1.7600.0
ServiceVersion     : 5.1.0.0
DatabaseUserName   : NT AUTHORITY\NETWORK SERVICE
Sid                : S-1-5-21-2316621082-1546847349-2782505528-1165
ActiveSiteServices : {
 MySiteService1, MySiteService2... }
```

Parameters

-Metadata

Gets records with matching metadata entries.

The value being compared with is a concatenation of the key name, a colon, and the value. For example: -Metadata “abc:x*”matches records with a metadata entry having a key name of “abc”and a value starting with the letter “x”.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Trust_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Trust_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Trust.Sdk.Service

The Get-TrustServiceInstance command returns an object containing the following properties.

Uid <Integer>

Specifies the unique identifier for the service in the group. The unique identifier is an index number.

ServiceHostId <Guid>

Specifies the unique identifier for the service instance.

DNSName <String>

Specifies the domain name of the host on which the service runs.

MachineName <String>

Specifies the short name of the host on which the service runs.

CurrentState <Citrix.Fma.Sdk.ServiceCore.ServiceState>

Specifies whether the service is running, started but inactive, stopped, or failed.

LastStartTime <DateTime>

Specifies the date and time at which the service was last restarted.

LastActivityTime <DateTime>

Specifies the date and time at which the service was last stopped or restarted.

OSType

Specifies the operating system installed on the host on which the service runs.

OSVersion

Specifies the version of the operating system installed on the host on which the service runs.

ServiceVersion

Specifies the version number of the service instance. The version number is a string that reflects the full build version of the service.

DatabaseUserName <string>

Specifies for the service instance the Active Directory account name with permissions to access the database. This will be either the machine account or, if the database is running on a controller, the NetworkService account.

Sid <string>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

ActiveSiteServices <string[]>

Specifies the names of active site services currently running in the service. Site services are components that perform long-running background processing in some services. This field is empty for services that do not contain site services.

Notes

If the command fails, the following errors can be returned.

Error Codes

UnknownObject

One of the specified objects was not found.

PartialData

Only a subset of the available data was returned.

InvalidFilter

A filtering expression was supplied that could not be interpreted for this cmdlet.

CouldNotQueryDatabase

The query required to get the database was not defined.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs

on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin

Get-TrustServiceAddedCapability

March 11, 2024

Gets any added capabilities for the Trust Service on the controller.

Syntax

```
Get-TrustServiceAddedCapability
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Enables updates to the Trust Service on the controller to be detected.

There is no requirement for a database connection to be configured in order for this command to be used.

Examples

EXAMPLE 1

Get the added capabilities of the Trust Service.

```
Get-TrustServiceAddedCapability
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

String containing added capabilities.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin

Get-TrustServiceInstance

March 11, 2024

Gets the service instance entries for the Trust Service.

Syntax

```
Get-TrustServiceInstance
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Returns service interfaces published by instances of the Trust Service. Each instance of a service publishes multiple interfaces with distinct interface types, and each of these interfaces is represented as a ServiceInstance object. Service instances can be used to register the service with a central configuration service so that other services can use the functionality.

You do not need to configure a database connection to use this command.

Examples

EXAMPLE 1

Get all instances of the Trust Service running on the specified machine. For remote services, use the AdminAddress parameter to define the service for which the interfaces are required. If the AdminAddress parameter has not been specified for the runspace, service instances running on the local machine are returned.

```
1  Get-TrustServiceInstance
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Trust.Sdk.ServiceInstance

The Get-TrustServiceInstance command returns an object containing the following properties.

ServiceGroupUid <Guid>

Specifies the unique identifier for the service group of which the service is a member.

ServiceGroupName <String>

Specifies the name of the service group of which the service is a member.

ServiceInstanceUID <Guid>

Specifies the unique identifier for registered service instances, which are service instances held by and obtained from a central configuration service. Unregistered service instances do not have unique identifiers.

ServiceType <String>

Specifies the service instance type. For this service, the service instance type is always Trust.

Address

Specifies the address of the service instance. The address can be used to access the service and, when registered in the central configuration service, can be used by other services to access the service.

Binding

Specifies the binding type that must be used to communicate with the service instance. In this release of XenDesktop, the binding type is always 'wcf_HTTP_kerb'. This indicates that the service provides a Windows Communication Foundation endpoint that uses HTTP binding with integrated authentication.

Version

Specifies the version of the service instance. The version number is used to ensure that the correct versions of the services are used for communications.

ServiceAccount <String>

Specifies the Active Directory account name for the machine on which the service instance is running. The account name is used to provide information about the permissions required for interservice communications.

ServiceAccountSid <String>

Specifies the Active Directory account security identifier for the machine on which the service instance is running.

InterfaceType <String>

Specifies the interface type. Each service can provide multiple service instances, each for a different purpose, and the interface defines the purpose. Available interfaces are:

SDK - for PowerShell operations

InterService - for operations between different services

Peer - for communications between services of the same type

Metadata <Citrix.Trust.Sdk.Metadata[]>

The collection of metadata associated with registered service instances, which are service instances held by and obtained from a central configuration service. Metadata is not stored for unregistered service instances.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustServiceStatus
- Reset-TrustServiceGroupMembership

Get-TrustServiceKey

March 11, 2024

Gets register ServiceKey configured for this site.

Syntax

```
Get-TrustServiceKey
    [[-Uid] <Int32>]
    [-InstanceId <String>]
    [-InstanceName <String>]
    [-IsTrustService <Boolean>]
    [-LastUpdated <DateTime>]
    [-RotationNeeded <Boolean>]
    [-ServiceName <String>]
    [-Property <String[]>]
    [-ReturnTotalRecordCount]
    [-MaxRecordCount <Int32>]
    [-Skip <Int32>]
    [-SortBy <String>]
    [-Filter <String>]
    [-FilterScope <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Retrieves registered ServiceKey matching the specified criteria. If no parameters are specified, all registered ServicesKey will be returned.

Examples

EXAMPLE 1

Gets all service keys.


```
Get-TrustServiceKey
```

EXAMPLE 2

Gets all service keys for a Trust Service.

```
Get-TrustServiceKey -IsTrustService $true
```

EXAMPLE 3

Mark all the service keys whose public key were last updated before October 20, 2015 to rotate their public keys.

```
Get-TrustServiceKey -Filter "LastUpdated -lt '10-20-2015'" | Set-
    TrustServiceKeyRotation
```

EXAMPLE 4

Revoke the previous public key of all service keys that were updated after October 20, 2016.

```
Get-TrustServiceKey -Filter "LastUpdated -gt '10-20-2016'" | Revoke-
    TrustPreviousServiceKey
```

Parameters

-Uid

Get ServiceKey with the specified Unique Identifier.

Type:	Int32
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InstanceId

Get ServiceKey with the specified instance identifier.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-InstanceName

Get ServiceKey with the specified instance name

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-IsTrustService

Get ServiceKey with the specified value of the IsTrustService flag.

If true, only Service Keys for the Trust Service will be returned.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LastUpdated

Get ServiceKey where the public key was last updated at the specified time.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RotationNeeded

Get ServiceKey with the specified value of the RotationNeeded flag.

If true, only ServiceKey that need to be rotated will be returned.

If false, only ServiceKey where key rotation is not pending will be returned.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceName

Get ServiceKey with the specified service name.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Trust_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Trust_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Trust.Sdk.ServiceKey

Get-TrustServiceKey returns an object for each matching registered Service Key.

Related Links

- about_TrustTrustSnapin
- Register-TrustServiceKey
- Set-TrustServiceKeyRotation
- Unregister-TrustServiceKey
- Revoke-TrustPreviousServiceKey

Get-TrustServiceStatus

March 11, 2024

Gets the current state of the Trust Service on the controller.

Syntax

```
Get-TrustServiceStatus
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Enables the status of the Trust Service on the controller to be determined. If the service has multiple data stores it will return the overall state as an aggregate of all the data store states. For example, if the site data store status is OK and the secondary data store status is DBUnconfigured then it will return DBUnconfigured. Before using this command, you don't have to configure the database connection to the Service.

Examples

EXAMPLE 1

Get the current status of the Trust Service.

```
Get-TrustServiceStatus
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Get-TrustServiceStatus command returns an object containing the status of the Trust Service together with extra diagnostics information.

DBUnconfigured

The Trust Service does not have a database connection configured.

DBRejectedConnection

The database rejected the logon attempt from the Trust Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

InvalidDBConfigured

The expected stored procedures are missing from the database. This may be because the Trust Service schema has not been added to the database.

DBNotFound

The specified database could not be located with the configured connection string.

DBMissingOptionalFeature

The Trust is connected to a database that is valid, but it does not have the full functionality required for optimal performance. Upgrading the database is advisable.

DBMissingMandatoryFeature

The Trust is connected to a database that is valid, but it does not have the full functionality required so the Trust cannot function. Upgrading the database is required.

DBNewerVersionThanService

The version of the Trust Service currently in use is incompatible with the version of the Trust Service schema on the database. Upgrade the Trust Service to a more recent version.

DBOlderVersionThanService

The version of the Trust Service schema on the database is incompatible with the version of the Trust Service currently in use. Upgrade the database schema to a more recent version.

DBVersionChangeInProgress

A database schema upgrade is currently in progress.

OK

The Trust Service is running and is connected to a database containing a valid schema.

PendingFailure

Connectivity between the Trust Service and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

Failed

Connectivity between the Trust and the database has been lost for an extended period of time, or has failed due to a configuration problem. The Trust service cannot operate while its connection to the database is unavailable.

Unknown

The service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Set-TrustDBConnection
- Test-TrustDBConnection
- Get-TrustDBConnection
- Get-TrustDBSchema

Get-TrustVdaEnrollmentToken

March 11, 2024

This cmdlet will get all the metadata records for all the VDA enrollment tokens.

Syntax

```
Get-TrustVdaEnrollmentToken
   [-Id <Guid>]
   [[-TokenName] <String>]
   [-CatalogId <Guid>]
   [-Property <String[]>]
   [-ReturnTotalRecordCount]
   [-MaxRecordCount <Int32>]
   [-Skip <Int32>]
   [-SortBy <String>]
   [-Filter <String>]
   [-FilterScope <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

This will not return the tokens themselves only the metadata used to create the tokens. The tokens are not stored after they are generated.

Examples

EXAMPLE 1

Get all the metadata records for all the VDA enrollment tokens.

```
Get-TrustVdaEnrollmentToken
```

Parameters

-TokenName

The name used to more easily identify this VDA enrollment token.

Type:	String
Position:	2
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	True

-Id

The JWT token ID of the VDA enrollment token meta data to retrieve.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-CatalogId

The Catalog Id that this VDA enrollment token allows machines to be added to.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Property

Specifies the properties to be returned. This is similar to piping the output of the command through Select-Object, but the properties are filtered more efficiently at the server.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ReturnTotalRecordCount

When specified, the cmdlet outputs an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. See about_Trust_Filtering for details.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxRecordCount

Specifies the maximum number of records to return.

Type:	Int32
Position:	Named
Default value:	250
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-Skip

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

Type:	Int32
Position:	Named
Default value:	0
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SortBy

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order. Ascending order is assumed if no prefix is present.

Type:	String
Position:	Named
Default value:	The default sort order is by name or unique identifier.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Filter

Gets records that match a PowerShell-style filter expression. See about_Trust_Filtering for details.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FilterScope

Gets only results allowed by the specified scope id.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Trust.Sdk.VdaEnrollmentToken[]

The vda enrollment token metadata that matches the provided filters

Related Links

New-TrustBearerToken

March 11, 2024

Creates a new bearer token for the current user.

Syntax

```
New-TrustBearerToken
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Creates a new bearer token for the current user. This uses the current Windows identity.

Examples

EXAMPLE 1

Obtains the bearer token and stores it in the $bearerData variable and uses it to authenticate with the Broker.

```
$bearerData = New-TrustBearerToken
New-BrokerDBConnection -BearerToken $bearerData.Token
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Trust.Sdk.BearerTokenData

The bearer token.

Related Links

- about_TrustTrustSnapin

New-TrustVdaEnrollmentToken

March 11, 2024

This cmdlet creates a new VDA enrollment token from the provided metadata.

Syntax

```
New-TrustVdaEnrollmentToken
   -CatalogId <Guid>
   [-ExpirationDate <DateTime>]
   [-HostConnectionId <Guid>]
   [-IssuedToUser <String>]
   [-NotValidBeforeDate <DateTime>]
   -NumMachinesAllowed <Int32>
   [-TokenName] <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
```



```
    [<CommonParameters>]
```

Description

VDA enrollment tokens are JWT tokens issued for limited use to add VDA machines to a machine catalog.

Examples

EXAMPLE 1

Creates a VDA enrollment token that allows the bearer to add up to 3 machines to machine catalog ID 30e1a941-cc17-4cc9-a4d7-2a49c7350430 that will expire on 4/1/2023.

```
New-TrustVdaEnrollmentToken -TokenName MyNewVdaToken -ExpirationDate
    4/1/2023 -NotValidBeforeDate 3/20/2023 -NumMachinesAllowed 3 -
    CatalogId 30e1a941-cc17-4cc9-a4d7-2a49c7350430
```

EXAMPLE 2

Creates a VDA registration token that allows the bearer to add up to 3 machines to machine catalog ID 30e1a941-cc17-4cc9-a4d7-2a49c7350430. Since no expiration date or not valid before date were provided this token will expire 14 days after the date it was created.

```
New-TrustVdaEnrollmentToken -TokenName MyNewVdaToken -
    NumMachinesAllowed 3 - CatalogId 30e1a941-cc17-4cc9-a4d7-2
    a49c7350430
```

EXAMPLE 3

Creates a VDA registration token that allows the bearer to add up to 3 machines to machine catalog ID 30e1a941-cc17-4cc9-a4d7-2a49c7350430 that is not valid before 7/4/2023 and will expire on 7/18/2023.

```
New-TrustVdaEnrollmentToken -TokenName MyNewVdaToken -
    NumMachinesAllowed 3 - CatalogId 30e1a941-cc17-4cc9-a4d7-2
    a49c7350430 -NotValidBeforeDate 7/4/2023
```


Parameters

-TokenName

The name used to more easily identify this VDA enrollment token.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-CatalogId

The machine catalog Id that this VDA enrollment token allows machines to be added to.

Type:	Guid
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-NumMachinesAllowed

The number of machines this VDA enrollment token allows to be added to the machine catalog.

Type:	Int32
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)

Accept wildcard characters:	False

-ExpirationDate

Optional - The UTC date this VDA enrollment token expires. This will default to creation date or NotValidBeforeDate plus the threshold of 14 days.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-HostConnectionId

The HostConnectionId that orchestration service can use to determine how to handle power management.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-IssuedToUser

Optional - The user to which this new VDA enrollment token has been issued.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-NotValidBeforeDate

Optional - The UTC date before which this VDA enrollment token will not be valid. This will default to the creation date.

Type:	DateTime
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Trust.Sdk.VdaEnrollmentToken

This token result object contains the VDA enrollment token and some additional metadata.

Related Links

Register-TrustServiceKey

March 11, 2024

Creates a new trusted Service Key

Syntax

```
Register-TrustServiceKey
        -ServiceName <String>
        [-InstanceId <String>]
        -PublicKey <String>
        [<CitrixCommonParameters>]
        [<CommonParameters>]
```


Description

Creates a new trusted Service Key. If this is a Service Key with the ServiceName of "ConnectorProxy", it will also result in adding of the corresponding Edge Server.

Examples

EXAMPLE 1

Register DCCHN-Proxy.xd.local ConnectorProxy with the Trust Service using the specified Public Key

```
Register-TrustServiceKey -ServiceName ConnectorProxy -InstanceId DCCHN-Proxy.xd.local -PublicKey PFJTQUtleVZhbHVlPjxNb2R1bHVzPnU2b0lCaFR2NXl...
```

Parameters

-ServiceName

The Name of the Service being effected

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PublicKey

The primary public key being registered.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-InstanceId

The instance ID of the service. This is usually the FQDN of the machine the service is running on.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Trust.Sdk.ServiceKey

The newly created Service Key.

Notes

This command should only be needed to register a ConnectorProxy that cannot do so using the installer and repair tool.

Related Links

- about_TrustTrustSnapin
- Get-TrustServiceKey
- Set-TrustServiceKeyRotation
- Unregister-TrustServiceKey
- Revoke-TrustPreviousServiceKey
- Get-ConfigEdgeServer

Remove-TrustServiceKeyMetadata

March 11, 2024

Removes metadata from the given ServiceKey.

Syntax

```
Remove-TrustServiceKeyMetadata
      [-ServiceKeyUid] <Int32>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-TrustServiceKeyMetadata
      [-ServiceKeyUid] <Int32>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-TrustServiceKeyMetadata
      [-InputObject] <ServiceKey[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```



```
Remove-TrustServiceKeyMetadata
       [-InputObject] <ServiceKey[]>
       -Map <PSObject>
       [-LoggingId <Guid>]
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given ServiceKey.

Examples

EXAMPLE 1

Remove all metadata from all ServiceKey objects.

```
Get-TrustServiceKey | % {
  Remove-TrustServiceKeyMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceKeyUid

Id of the ServiceKey

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceKey[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Set-TrustServiceKeyMetadata

Remove-TrustServiceMetadata

March 11, 2024

Removes metadata from the given Service.

Syntax

```
Remove-TrustServiceMetadata
      [-ServiceHostId] <Guid>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-TrustServiceMetadata
      [-ServiceHostId] <Guid>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-TrustServiceMetadata
      [-InputObject] <Service[]>
      -Name <String>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Remove-TrustServiceMetadata
      [-InputObject] <Service[]>
      -Map <PSObject>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Provides the ability to remove metadata from the given Service.

Examples

EXAMPLE 1

Remove all metadata from all Service objects.


```
Get-TrustService | % {
  Remove-TrustServiceMetadata -Map $_.MetadataMap  }
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be removed.

Type:	Service[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

The metadata property to remove.

Type:	String
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can be either a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”). The properties whose names match keys in the map will be removed.

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Set-TrustServiceMetadata

Remove-TrustVdaEnrollmentToken

March 11, 2024

This cmdlet will delete the specified VDA enrollment token.

Syntax

```
Remove-TrustVdaEnrollmentToken
      [-InputObject] <VdaEnrollmentToken[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
```



```
5       [<CommonParameters>]
```

```
1 Remove-TrustVdaEnrollmentToken
2       [-Id] <Guid[]>
3       [-LoggingId <Guid>]
4       [<CitrixCommonParameters>]
5       [<CommonParameters>]
```

```
1 Remove-TrustVdaEnrollmentToken
2       [-TokenName] <String[]>
3       [-LoggingId <Guid>]
4       [<CitrixCommonParameters>]
5       [<CommonParameters>]
```

Description

This cmdlet will delete the specified VDA enrollment token metadata from the database making the token useless.

Examples

EXAMPLE 1

Deletes the VDA enrollment token metadata with an ID of 00f1f4e4-ce0c-4255-ae81-17d631d23eb7.

```
1 Remove-TrustVdaEnrollmentToken -Id 00f1f4e4-ce0c-4255-ae81-17d631d23eb7
```

Parameters

-InputObject

Specifies the VDA enrollment token meta data objects to delete

Type:	VdaEnrollmentToken[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Id

The JWT token ID of the VDA enrollment token to delete.

Type:	Guid[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-TokenName

The TokenName of the VDA enrollment token to delete.

Type:	String[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-TrustEnabledFeatureList

March 11, 2024

Refreshes the Trust service's list of enabled features.

Syntax

```
Reset-TrustEnabledFeatureList
     [<CitrixCommonParameters>]
     [<CommonParameters>]
```

Description

Synchronizes the currently selected Citrix Trust Service instance's list of enabled features with that held by the Central Configuration Service.

By default toggling site features on or off can take many minutes to propagate to all services in the site. Howerver, after a toggle is changed, if this cmdlet is run for each service instance in the site the changes propagate effectively immediately.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

Examples

EXAMPLE 1

Refreshes the selected Trust service instance's list of enabled features.

```
Reset-TrustEnabledFeatureList
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Reset-TrustServiceGroupMembership

March 11, 2024

Reloads the access permissions and configuration service locations for the Trust Service.

Syntax

```
Reset-TrustServiceGroupMembership
      [-ConfigServiceInstance] <ServiceInstance[]>
      [-LoggingId <Guid>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Enables you to reload Trust Service access permissions and configuration service locations. The Reset-TrustServiceGroupMembership command must be run on at least one instance of the service type (Trust) after installation and registration with the configuration service. Without this operation, the Trust services will be unable to communicate with other services in the XenDesktop deployment. When the command is run, the services are updated when additional services are added to the deployment, provided that the configuration service is not stopped. The Reset-TrustServiceGroupMembership command can be run again to refresh this information if automatic updates do not occur when new services are added to the deployment. If more than one configuration service instance is passed to the command, the first instance that meets the expected service type requirements is used.

Examples

EXAMPLE 1

Reset the service group membership for a service in a deployment where the configuration service is configured and running on the same machine as the service.

```
Get-ConfigRegisteredServiceInstance -ServiceType Config | Reset-
    TrustServiceGroupMembership
```

EXAMPLE 2

Reset the service group membership for a service in a deployment where the configuration service that is configured and running on a machine named 'OtherServer.example.com'.

```
Get-ConfigRegisterdServiceInstance -ServiceType Config -AdminAddress
    OtherServer.example.com | Reset-TrustServiceGroupmembership
```

Parameters

-ConfigServiceInstance

Specifies the configuration service instance object that represents the service instance for the type 'InterService'that references a configuration service for the deployment.

Type:	ServiceInstance[]
Position:	2
Default value:	LocalHost
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Trust.Sdk.ServiceInstance[]

Service instances containing a ServiceInstance object that refers to the central configuration service interservice interface can be piped to the Reset-TrustServiceGroupMembership command.

Outputs

Citrix.Trust.Sdk.ServiceInstance

Reset-TrustServiceGroupMembership returns opaque objects containing Configuration Service instances that are used by the Trust Service instance.

Notes

If the command fails, the following errors can be returned.

Error Codes

NoSuitableServiceInstance

None of the supplied service instance objects were suitable for resetting service group membership.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustServiceInstance
- Get-TrustServiceStatus

Revoke-TrustBearerToken

March 11, 2024

Revokes the user's current bearer token.

Syntax

```
Revoke-TrustBearerToken
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Revokes the user's current bearer token.

Examples

EXAMPLE 1

Revokes any bearer token issued for the current user.

```
Revoke-TrustBearerToken
```

Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_TrustTrustSnapin

Revoke-TrustPreviousServiceKey

March 11, 2024

This cmdlet removes the older public key from the service key

Syntax

```
Revoke-TrustPreviousServiceKey
      -ServiceName <String>
      [-InstanceId <String>]
      [-PassThru]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

```
Revoke-TrustPreviousServiceKey
      -ServiceKey <ServiceKey>
      [-PassThru]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

This cmdlet removes the older public key from the service key

Examples

EXAMPLE 1

Revoke the previous public key for the DCCHN-Proxy.xd.local ConnectorProxy service.


```
Revoke-TrustPreviousServiceKey -ServiceName ConnectorProxy -InstanceId
    DCCHN-Proxy.xd.local
```

EXAMPLE 2

Revoke the previous public key of all service keys that were updated after October 20, 2016.

```
Get-TrustServiceKey -Filter "LastUpdated -gt '10-20-2016'" | Revoke-
    TrustPreviousServiceKey
```

Parameters

-ServiceName

The Name of the service being effected

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceKey

An existing Service key. Usually the results piped from another command.

Type:	ServiceKey
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InstanceId

The instance ID of the service. This is usually the FQDN of the machine the service is running on.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Trust.Sdk.ServiceKey

You can pipe the service keys to be updated into this command.

Outputs

None or Citrix.Trust.Sdk.ServiceKey

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a ServiceKey object.

Related Links

- about_TrustTrustSnapin
- Register-TrustServiceKey
- Set-TrustServiceKeyRotation
- Get-TrustServiceKey
- Unregister-TrustServiceKey

Revoke-TrustVdaEnrollmentToken

March 11, 2024

This cmdlet will mark the specified VDA enrollment token as revoked.

Syntax

```
Revoke-TrustVdaEnrollmentToken
       [-TokenId] <String>
       [<CitrixCommonParameters>]
       [<CommonParameters>]
```

Description

If the token is revoked it will no longer be valid. Revoked tokens will eventually be deleted.

Examples

EXAMPLE 1

Marks the VDA enrollment token with an ID of 00f1f4e4-ce0c-4255-ae81-17d631d23eb7 as revoked.

```
Revoke-TrustVdaEnrollmentToken -TokenId 00f1f4e4-ce0c-4255-ae81-17
    d631d23eb7
```

Parameters

-TokenId

The JWT token ID of the VDA enrollment token to revoke.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

Set-TrustDBConnection

March 11, 2024

Configures a database connection for the Trust Service.

Syntax

```
Set-TrustDBConnection
   [-DBConnection] <String>
   [-Force]
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies the database connection string for use by the currently selected Citrix Trust Service instance.

The service records the connection string and attempts to contact the specified database. If the database cannot initially be contacted the service retries the connection at intervals until contact with the database is successfully established.

Is is not possible to set a new database connection string for the service if one is already recorded. The connection string must first be set to $null. This action causes the service to reset and return to its idle state, at which point a new connection string can be set.

When a database connection string is successfully set for the service, a status of OK is returned by the cmdlet. If the database connection string is set to $null, a DBUnconfigured status is returned.

A syntactically invalid connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

Examples

EXAMPLE 1

Configures the service instance to use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Set-TrustDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
    Trusted_Connection=True"
```

EXAMPLE 2

Resets the service instance's database connection string. The service instance resets and returns to an idle state until a valid new database connection string is specified.

```
Set-TrustDBConnection -DBConnection $null
```

Parameters

-DBConnection

Specifies the database connection string to be used by the Trust Service. Passing in $null will clear any existing database connection configured.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Force

If present, allows the local administrator to set the connection string to null when there are problems contacting the database or other services.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Set-TrustDBConnection cmdlet returns an object describing the status of the Trust Service together with extra diagnostics information. Possible values are:

- OK:

The Trust Service instance is configured with a valid database and service schema. The service is operational.

- DBUnconfigured:

No database connection string is set for the Trust Service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Trust Service. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Trust Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the configured connection string.

- DBNewerVersionThanService:

The version of the Trust Service currently in use is newer than, and incompatible with, the version of the Trust Service schema on the database. Upgrade the Trust Service to a more recent version.

- DBOlderVersionThanService:

The version of the Trust Service schema on the database is newer than, and incompatible with, the version of the Trust Service currently in use. Upgrade the database schema to a more recent version.

- DBVersionChangeInProgress:

A database schema upgrade is in progress.

- PendingFailure:

Connectivity between the Trust Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Trust Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

The status of the Trust Service cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

DatabaseConnectionDetailsAlreadyConfigured

There was already a database connection configured.

After a configuration is set, it can only be set to $null.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustServiceStatus
- Get-TrustDBConnection
- Test-TrustDBConnection

Set-TrustDBCredentials

March 11, 2024

Configures the database server SQL credentials for the Trust Service.

Syntax

```
Set-TrustDBCredentials
   [-Credentials] <PSCredential>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustDBCredentials
   [-Login] <String>
   [-Password] <SecureString>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Specifies SQL credentials to be used by the currently selected Citrix Trust Service instance to authenticate with the database server. By default Windows authentication is used and no SQL credentials are required.

If used, SQL credentials must be specified before the service's schema is obtained and created, and before the database connection string is set. The credentials must also be specified for each additional Trust Service prior to it being added to the site.

If the database connection string is already set and Windows authentication is in use, it is not possible to specify SQL credentials, however, if SQL credentials are already in use then they can be changed.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

Examples

EXAMPLE 1

Prompts for SQL credentials and sets them for use by the current service instance.

```
Set-TrustDBCredentials
```

EXAMPLE 2

Prompts for SQL credentials and sets them for use by the current service instance. This form is useful where the same credentials are being used multiple times.

```
$sqlCred = Get-Credential
Set-TrustDBCredentials $sqlCred
```

EXAMPLE 3

Sets the SQL credentials to the explicitly specified login and password values.

```
$password = ConvertTo-SecureString 'P@ssW0rd' -AsPlainText -Force
Set-TrustDBCredentials 'CvadLogin' $password
```

EXAMPLE 4

Clears previously set SQL credentials and reverts to use of the default Windows authentication. This can only be done if no connection string is set.

```
Set-TrustDBCredentials $null
```


Parameters

-Credentials

A PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

The SQL login to be used for SQL authentication.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

The password to be used for SQL authentication.

Type:	SecureString
Position:	3
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- Get-TrustDBSchema
- Set-TrustDBConnection
- Get-Credential

Set-TrustServiceKeyMetadata

March 11, 2024

Adds or updates metadata on the given ServiceKey.

Syntax

```
Set-TrustServiceKeyMetadata
   [-ServiceKeyUid] <Int32>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustServiceKeyMetadata
   [-ServiceKeyUid] <Int32>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustServiceKeyMetadata
   [-InputObject] <ServiceKey[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```



```
Set-TrustServiceKeyMetadata
   [-InputObject] <ServiceKey[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

Description

Provides the ability for additional custom data to be stored against given ServiceKey objects.

Examples

EXAMPLE 1

Add metadata with a name of 'property'and a value of 'value'to the ServiceKey with the identifier '4CECC26E-48E1-423F-A1F0-2A06DDD0805C'.

```
Set-TrustServiceKeyMetadata -ServiceKeyUid 4CECC26E-48E1-423F-A1F0-2
   A06DDD0805C -Name property -Value value

Key                                          Value
---                                          -----
property                                     value
```

Parameters

-ServiceKeyUid

Id of the ServiceKey

Type:	Int32
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which the metadata is to be added.

Type:	ServiceKey[]
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the ServiceKey specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{“name1”= “val1”; “name2”= “val2”}) or a string dictionary (created with new-object “System.Collections.Generic.Dictionary[String,String]”).

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-TrustServiceKeyMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Remove-TrustServiceKeyMetadata

Set-TrustServiceKeyRotation

March 11, 2024

Mark a service key as needing to be rotated

Syntax

```
Set-TrustServiceKeyRotation
   -ServiceName <String>
   [-InstanceId <String>]
   [-PassThru]
```



```
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Set-TrustServiceKeyRotation
   -ServiceKey <ServiceKey>
    [-PassThru]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Mark a service key as needing to be rotated. Key rotation will not happen immediately. Typically, it should take about an hour for the keys to be rotated but it can take longer if the service that owns the key is down or unable to contact the Trust Service.

Use Get-TrustServiceKey and look at the "LastUpdated"and "RotationNeeded"fields to determine when it was last rotated and if the Service Key has already been marked for Rotation.

Examples

EXAMPLE 1

Mark the Service Key for the DCCHN-Proxy.xd.local ConnectorProxy to be rotated.

```
Set-TrustServiceKeyRotation -ServiceName ConnectorProxy -InstanceId
   DCCHN-Proxy.xd.local
```

EXAMPLE 2

Mark all the Service Keys that were last updated before October 20, 2015 to rotate their public keys.

```
Get-TrustServiceKey -Filter "LastUpdated -lt '10-20-2015'" | Set-
   TrustServiceKeyRotation
```

Parameters

-ServiceName

The Name of the Service being effected

Type: String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceKey

An existing Service key. Usually the results piped from another command.

Type:	ServiceKey
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InstanceId

The instance ID of the service. This is usually the FQDN of the machine the service is running on.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Trust.Sdk.ServiceKey

You can pipe the service keys to be updated into this command.

Outputs

None or Citrix.Trust.Sdk.ServiceKey

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a ServiceKey object.

Related Links

- about_TrustTrustSnapin
- Register-TrustServiceKey

- Get-TrustServiceKey
- Unregister-TrustServiceKey
- Revoke-TrustPreviousServiceKey

Set-TrustServiceMetadata

March 11, 2024

Adds or updates metadata on the given Service.

Syntax

```
Set-TrustServiceMetadata
   [-ServiceHostId] <Guid>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustServiceMetadata
   [-ServiceHostId] <Guid>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustServiceMetadata
   [-InputObject] <Service[]>
   -Name <String>
   -Value <String>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```

```
Set-TrustServiceMetadata
   [-InputObject] <Service[]>
   -Map <PSObject>
   [-LoggingId <Guid>]
   [<CitrixCommonParameters>]
   [<CommonParameters>]
```


Description

Allows you to store additional custom data against given Service objects.

Examples

EXAMPLE 1

Add metadata with a name of ‘property’and a value of ‘value’to the Service with the identifier ‘4CECC26E-48E1-423F-A1F0-2A06DDD0805C’.

```
Set-TrustServiceMetadata -ServiceHostId 4CECC26E-48E1-423F-A1F0-2
    A06DDD0805C -Name property -Value value

Key                                    Value
---                                    -----
property                               value
```

Parameters

-ServiceHostId

Id of the Service

Type:	Guid
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName, ByValue)
Accept wildcard characters:	False

-InputObject

Objects to which metadata is to be added.

Type:	Service[]
Position:	2
Default value:	None

Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-Name

Specifies the property name of the metadata to be added. The property must be unique for the Service specified. The property cannot contain any of the following characters \/;:#.*?=<>|[]()'"

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Value

Specifies the value for the property.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Map

Specifies a dictionary of (name, value)-pairs for the properties. This can either be a hashtable (created with @{"name1"= "val1"; "name2"= "val2"}) or a string dictionary (created with new-object "System.Collections.Generic.Dictionary[String,String]").

Type:	PSObject
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

System.Collections.Generic.Dictionary[String,String]

Set-TrustServiceMetadata returns a dictionary containing the new (name, value)-pairs.

- Key <string>

  Specifies the name of the property.

- Value <string>

  Specifies the value for the property.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidParameterCombination

The cmdlet parameters are inconsistent.

UnknownObject

One of the specified objects was not found.

DatabaseError

An error occurred in the service while attempting a database operation.

DatabaseNotConfigured

The operation could not be completed because the database for the service is not configured.

DataStoreException

An error occurred in the service while attempting a database operation - communication with the database failed for various reasons.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Remove-TrustServiceMetadata

Test-TrustDBConnection

March 11, 2024

Tests whether a database is suitable for use by the Citrix Trust Service.

Syntax

```
Test-TrustDBConnection
    [-DBConnection] <String>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

```
Test-TrustDBConnection
    [-DBConnection] <String>
    [-Credentials] <PSCredential>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```



```
Test-TrustDBConnection
    [-DBConnection] <String>
    [-Login] <String>
    [-Password] <SecureString>
    [-LoggingId <Guid>]
    [<CitrixCommonParameters>]
    [<CommonParameters>]
```

Description

Tests whether the database specified in the given connection string is suitable for use by the currently selected Citrix Trust Service instance.

The service attempts to contact the specified database and returns a status indicating whether the database is both contactable and usable. The test does not impact any currently established connection from the service instance to another database in any way. The tested connection string is not recorded.

Only use of Windows authentication within the connection string is supported; a connection string containing SQL authentication credentials is always rejected as invalid.

The current service instance is the one on the local machine, or the one most recently specified using the -AdminAddress parameter of a Trust SDK cmdlet.

Examples

EXAMPLE 1

Tests whether the service instance could use a database called XDDB on an SQL Server Express database running on the machine called dbserver. Integrated Windows authentication is required.

```
Test-TrustDBConnection "Server=dbserver\SQLEXPRESS;Database=XDDB;
   Trusted_Connection=True"
```

Parameters

-DBConnection

Specifies the database connection string to be tested by the currently selected Citrix Trust Service instance.

Type: String

Position:	1
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Credentials

If using SQL authentication, a PSCredential object containing the SQL credentials to be used. This can be created using the Get-Credential cmdlet. The credentials required are the SQL server login and its associated password. By default Windows authentication is used and no credentials need to be specified.

Type:	PSCredential
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Login

If using SQL authentication, the SQL server login to use. By default Windows authentication is used and no login needs to be specified.

Type:	String
Position:	2
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Password

If using SQL authentication, the SQL server password to use. By default Windows authentication is used and no password needs to be specified.

Type:	SecureString
Position:	3
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingId

Specifies the identifier of the high-level operation this cmdlet call forms a part of. Citrix Studio and Director typically create high-level operations. PowerShell scripts can also wrap a series of cmdlet calls in a high-level operation by way of the Start-LogHighLevelOperation and Stop-LogHighLevelOperation cmdlets.

Type:	Guid
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -

WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Citrix.Fma.Sdk.Utilities.Service.ServiceStatusInfo

The Test-TrustDBConnection cmdlet returns an object describing the status of the selected Trust Service instance that would result if the connection string were used with the Set-TrustDBConnection cmdlet together with extra diagnostics information for the specified connection string. The actual current status of the service is not changed. Possible diagnostic values are:

- OK:

The Set-TrustDBConnection command would succeed if it were executed with the supplied connection string.

- DBUnconfigured:

No database connection string is set for the service instance.

- DBRejectedConnection:

The database rejected the logon attempt from the Trust Service instance. This may be because the service attempted to log on with invalid credentials or because a database has not been installed in the specified location.

- InvalidDBConfigured:

The specified database does not exist, is not visible to the Trust Service instance, or the service's schema within the database is invalid.

- DBNotFound:

The specified database could not be located with the given connection string.

- DBNewerVersionThanService:

The Trust Service instance is older than, and incompatible with, the service's schema in the database. The service instance needs upgrading.

- DBOlderVersionThanService:

The Trust Service instance is newer than, and incompatible with, the service's schema in the database. The database schema needs upgrading.

- DBVersionChangeInProgress:

A database schema upgrade is currently in progress.

- PendingFailure:

Connectivity between the Trust Service instance and the database has been lost. This may be a transitory network error, but may indicate a loss of connectivity that requires administrator intervention.

- Failed:

Connectivity between the Trust Service instance and the database has been lost for an extended period of time, or has failed due to a configuration problem. The service instance cannot operate while its connection to the database is unavailable.

- Unknown:

Service status cannot be determined.

Notes

If the command fails, the following errors can be returned.

Error Codes

InvalidDBConnectionString

The database connection string has an invalid format.

PermissionDenied

You do not have permission to execute this command.

AuthorizationError

There was a problem communicating with the Citrix Delegated Administration Service.

ConfigurationLoggingError

The operation could not be performed because of a configuration logging error.

CommunicationError

There was a problem communicating with the remote service.

ExceptionThrown

An unexpected error occurred. For more details, see the Windows event logs on the controller or the XenDesktop logs.

Related Links

- about_TrustTrustSnapin
- Get-TrustServiceStatus
- Get-TrustDBConnection
- Set-TrustDBConnection

Unregister-TrustServiceKey

March 11, 2024

Unregister and Revoke the service keys for a specified service from the site.

Syntax

```
Unregister-TrustServiceKey
           -ServiceName <String>
           [-InstanceId <String>]
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```

```
Unregister-TrustServiceKey
           -ServiceKey <ServiceKey>
           [<CitrixCommonParameters>]
           [<CommonParameters>]
```

Description

Unregister and Revoke the service keys for a specified service. If this is a service key with the Service-Name of “ConnectorProxy”, it will also result in the removal of the corresponding Edge Server.

Examples

EXAMPLE 1

Remove the service key for the ConnectorProxy DCCHN-Proxy.xd.local

```
1 Unregister-TrustServiceKey -ServiceName ConnectorProxy -InstanceId
     DCCHN-Proxy.xd.local
```

EXAMPLE 2

Unregister all the service keys associated with DCCHN-DDC1.xd.local

```
1 Get-TrustServiceKey -InstanceId DCCHN-DDC1.xd.local | Unregister-
     TrustServiceKey
```

Parameters

-ServiceName

The Name of the Service being effected

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ServiceKey

An existing Service key. Usually the results piped from another command.

Type:	ServiceKey
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-InstanceId

The instance ID of the service. This is usually the FQDN of the machine the service is running on.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Citrix.Trust.Sdk.ServiceKey

You can pipe the service keys to be updated into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_TrustTrustSnapin
- Register-TrustServiceKey
- Get-TrustServiceKey
- Set-TrustServiceKeyRotation
- Revoke-TrustPreviousServiceKey
- Get-ConfigEdgeServer

Update-TrustBearerToken

March 11, 2024

Requests a renewed bearer token.

Syntax

```
Update-TrustBearerToken
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Requests a renewed bearer token.

Examples

EXAMPLE 1

Obtains a bearer token for the current user and immediately refreshes it.

```
$bearerData = New-TrustBearerToken
Update-TrustBearerToken -BearerToken $bearerData.Token
```


Parameters

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

You can pipe the bearer token string into this command.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_TrustTrustSnapin

Update-TrustServiceKey

March 11, 2024

Update\Rotate the public key of a service

Syntax

```
Update-TrustServiceKey
      -ServiceName <String>
      [-InstanceId <String>]
      -PublicKey <String>
      [-InstanceName <String>]
      [<CitrixCommonParameters>]
      [<CommonParameters>]
```

Description

Updates primary public key for a given service and (optional) instance id, rotating keys accordingly.

Use Get-TrustServiceKey and look at the “LastUpdated”and “RotationNeeded”fields to determine when it was last rotated and if the Service Key needs to be rotated.

Examples

EXAMPLE 1

Rotate the Service Key for the DCCHN-Proxy.xd.local ConnectorProxy.

```
Update-TrustServiceKey -ServiceName ConnectorProxy -InstanceId DCCHN-
   Proxy.xd.local -PublicKey "newKey" -InstanceName "Instance1"
```

Parameters

-ServiceName

The Name of the Service being updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PublicKey

A new Service key.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-InstanceId

The instance ID of the service.

Type:	String
Position:	Named
Default value:	$null
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-InstanceName

The name of the instance for this service.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CitrixCommonParameters

This cmdlet supports the common Citrix parameters: -AdminAddress, -AdminClientIP, -BearerToken, -TraceParent, -TraceState and -VirtualSiteId. For more information, see about_CitrixCommonParameters.

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None or Citrix.Trust.Sdk.ServiceKey

This cmdlet returns a ServiceKey object.

Related Links

- about_TrustTrustSnapin
- Register-TrustServiceKey
- Get-TrustServiceKey
- Unregister-TrustServiceKey
- Revoke-TrustPreviousServiceKey

about_UpmSnapInUserProfileManagerSnapin

March 11, 2024

Topic

about_UpmSnapInUserProfileManagerSnapin

Short Description

This snapin is used to administer Citrix User Profile Management.

Command Prefix

All commands in this snap-in have the noun prefixed with 'UserProfile'.

Long Description

This SDK contains commands for configuration and profile store tests.

about_UpmSnapInUserProfileManagerSnapin

March 11, 2024

Topic

about_UpmSnapInUserProfileManagerSnapin

Short Description

This snapin is used to administer Citrix User Profile Management.

Command Prefix

All commands in this snap-in have the noun prefixed with 'UserProfile'.

Long Description

This SDK contains commands for configuration and profile store tests.

about_UserProfileManager_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
    PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results. Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:


```
# Escape examples
Get-<Noun> -Company "Abc[*]"                    # Matches Abc*
Get-<Noun> -Company "Abc`*"                     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}

Get-<Noun> -Filter { Shape -like 'C*'}

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

Get-<Noun> -Filter { StartTime -ge "2010-08-23T12:30:00.0Z"}
$d = [DateTime]"2010-08-23T12:30:00.0Z"
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

Get-<Noun> -Filter { StartTime -ge '-2'} # Two days ago
Get-<Noun> -Filter { StartTime -ge '-1:30'} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge '-0:0:30'} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*"}
Get-<Noun> -Filter { Users -contains "Fred*"}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
```



```
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq ‘Circle’-or Shape -eq ‘Square’}
$shapes = ‘Circle’,’Square’
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq ‘Blue’-and Shape -eq ‘Circle’) }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{


```
break;
}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

Get-UserProfileDefinition

March 11, 2024

Gets the high level description of a configuration for profile management, based on a byte array (blob).

Syntax

```
Get-UserProfileDefinition
   [-ByteArray] <Byte[]>
   [-ExcludeUnconfigured]
   [<CommonParameters>]
```

Description

Use this command to convert a configuration byte array into a set of named property settings. The byte array will either have been retrieved from the Broker, or from the New-UserProfileConfiguration cmdlet.

Examples

EXAMPLE 1

The first command creates a fresh configuration set in its default state, and stores it in a Windows PowerShell variable. The second command interprets the new blob, and would output the individual

properties of the default configuration.

```
$blob = New-UserProfileConfiguration

Get-UserProfileDefinition -ByteArray $blob
```

EXAMPLE 2

This command creates a fresh configuration set in its default state, and pipes the resulting byte array through to Get-UserProfileDefinition, which will interpret it and output its individual properties.

```
New-UserProfileConfiguration | Get-UserProfileDefinition
```

Parameters

-ByteArray

Specifies the low-level byte array (blob) to be interpreted.

Type:	Byte[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ExcludeUnconfigured

When this switch is supplied, the returned object will only carry the properties whose values have been explicitly set. Properties whose values are unconfigured will be omitted. When this switch is not supplied, the full set of configuration properties will be returned, and any unconfigured settings will just appear to be set to their default value.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

Byte[]

The cmdlet accepts the ByteArray parameter as pipeline input.

Outputs

PSObject

This cmdlet outputs a single PSObject with multiple properties. Each property denotes one setting within the configuration set. The following is a summary of the properties and their data types:

ServiceActive (bool?)

ProcessedGroups (string[])

ExcludedGroups (string[])

ProcessAdmins (bool?)

UserStorePath (string)

PSMidSessionWriteBack (bool?)

OfflineSupport (bool?)

DeleteCachedProfilesOnLogoff (bool?)

ProfileDeleteDelay (int?)

MigrateWindowsProfilesToUserStore (MigrateWindowsProfilesToUserStoreEnum?)

LocalProfileConflictHandling (LocalProfileConflictHandlingEnum?)

TemplateProfilePath (string)

TemplateProfileOverridesLocalProfile (bool?)

TemplateProfileOverridesRoamingProfile (bool?)

TemplateProfileIsMandatory (bool?)

LoadRetries (int?)

ProcessCookieFiles (bool?)

DisableDynamicConfig (bool?)

LogoffRatherThanTempProfile (bool?)

DebugMode (bool?)

LogLevel_Warnings (bool?)

LogLevel_Information (bool?)

LogLevel_FileSystemNotification (bool?)

LogLevel_FileSystemActions (bool?)

LogLevel_RegistryActions (bool?)

LogLevel_RegistryDifference (bool?)

LogLevel_ActiveDirectoryActions (bool?)

LogLevel_PolicyUserLogon (bool?)

LogLevel_Logon (bool?)

LogLevel_Logoff (bool?)

LogLevel_UserName (bool?)

MaxLogSize (int?)

DebugFilePath (string)

ExclusionList (string[])

IncludeListRegistry (string[])

ExclusionListSyncFiles (string[])

ExclusionListSyncDir (string[])

SyncDirList (string[])

SyncFileList (string[])

MirrorFoldersList (string[])

PSEnabled (bool?)

PSAlwaysCache_Enabled (bool?)

PSAlwaysCache (int?)

PSPendingLockTimeout (int?)

PSUserGroups (string[])

CPEnable (bool?)

CPUserGroups (string[])

CPSchemaPathData (string)

CPPathData (string)

CPMigrationFromBaseProfileToCPStore (bool?)

FRAdminAccess (bool?)

FRIncDomainName (bool?)

FRAppDataEnabled (FRAppDataEnum?)

FRAppDataPath (string)

FRDesktopEnabled (FRDesktopEnum?)

FRDesktopPath (string)

FRStartMenuEnabled (FRStartMenuEnum?)

FRStartMenuPath (string)

FRDocumentsEnabled (FRDocumentsEnum?)

FRDocumentsPath (string)

FRPicturesEnabled (FRPicturesEnum?)

FRPicturesPath (string)

FRMusicEnabled (FRMusicEnum?)

FRMusicPath (string)

FRVideosEnabled (FRVideosEnum?)

FRVideosPath (string)

FRFavoritesEnabled (FRFavoritesEnum?)

FRFavoritesPath (string)

FRContactsEnabled (FRContactsEnum?)

FRContactsPath (string)

FRDownloadsEnabled (FRDownloadsEnum?)

FRDownloadsPath (string)

FRLinksEnabled (FRLinksEnum?)

FRLinksPath (string)

FRSearchesEnabled (FRSearchesEnum?)

FRSearchesPath (string)

FRSavedGamesEnabled (FRSavedGamesEnum?)

FRSavedGamesPath (string)

Related Links

- about_UpmSnapInUserProfileManagerSnapin

Get-UserProfileManagerServiceAddedCapability

March 11, 2024

Lists the properties that can be configured through user profile configuration.

Syntax

```
Get-UserProfileManagerServiceAddedCapability
   [<CommonParameters>]
```

Description

This command produces an array containing the names of the properties that can be configured as part of a user profile configuration. Any name that occurs in the array can be used as a parameter for the Set-UserProfileDefinition cmdlet.

Examples

EXAMPLE 1

This command creates a list of all the properties that can be set through Set-UserProfileDefinition and stores them in $properties.


```
$properties = Get-UserProfileManagerServiceAddedCapability
```

Parameters

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this command

Outputs

Object[]

The names of the properties that can be configured through user profile configurations

Related Links

- about_UpmSnapInUserProfileManagerSnapin
- New-UserProfileConfiguration
- Set-UserProfileDefinition

Get-UserProfilePath

March 11, 2024

Gets the path (or paths) at which an individual user's profile is stored.

Syntax

```
Get-UserProfilePath
   -User <String>
   -PathToUserStore <String>
   [<CommonParameters>]
```

Description

Use this cmdlet to search for specific user profile folders, based on a configured store path.

The cmdlet will perform automatic substitutions on all of the dynamic parts of the store path. These substitutions will be made based either on the details of the individual user given, or by enumerating all possible strings that are permitted by the variable in question. The approach taken depends on the variable. User-specific variables such as %USERNAME% will be substituted based on the user. Variables such as !CTX_OSNAME! will be substituted by enumerating the possibilities.

Once all substitutions have been made, the cmdlet will search the user store for any directories that exist, matching the substituted path. All such directories are then reported as outputs of the cmdlet.

Once this cmdlet has been used to locate the profile store, other Windows PowerShell scripts can be written to inspect or manipulate the contents of the profile, provided that such access is allowed by the permissions of the profile store.

Examples

EXAMPLE 1

This command attempts to locate the user profile folder for the user account Fred within the domain FABRIKAM, where the user has been specified as an NT account name. This user's details will be substituted into the profile store path. A likely return result would be \\server\Profiles$\Fred.FABRIKAM.

```
Get-UserProfilePath -PathToUserStore \\server\Profiles$\%USERNAME%.%
    USERDOMAIN% -User FABRIKAM\Fred
```

EXAMPLE 2

This command attempts to locate the user profile folder for the user account Fred within the domain FABRIKAM, where the user has been specified as a User Principal Name (UPN). This user's details will be substituted into the profile store path. Possible return results might be \\server\Profiles$\Fred.FABRIKAM\Win7 or \\server\Profiles$\Fred.FABRIKAM\WinXP.


```
Get-UserProfilePath -PathToUserStore \\server\Profiles$\%USERNAME%.%
    USERDOMAIN%\!CTX_OSNAME! -User Fred@FABRIKAM
```

EXAMPLE 3

This command attempts to locate the user profile folder for the user account whose Security Identifier (SID) has been specified. This user's details will be substituted into the profile store path. Possible return results might be \\server\Profiles$\Fred.FABRIKAM\x86 or \\server\Profiles$\Fred.FABRIKAM\x64.

```
Get-UserProfilePath -PathToUserStore \\server\Profiles$\%USERNAME%.%
    USERDOMAIN%\!CTX_OSBITNESS! -User S
    -1-5-32-1045337234-12924708993-5683276719-19000
```

Parameters

-User

Specifies the user whose profile is to be found within the store. Users can be identified in terms of their account name, UPN or SID.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	True

-PathToUserStore

Specifies the UNC path to the configured profile store.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	True

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

String

The pipeline accepts any valid User parameter, as an account name, UPN or SID.

Outputs

String

The UNC path to the profile folder(s) belonging to the specified user. Each reported path will denote a directory that exists within the store, and will have all of its dynamic elements substituted with static strings.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

New-UserProfileConfiguration

March 11, 2024

Creates a new configuration for Citrix Profile Management, with all settings in their default initial state.

Syntax

```
New-UserProfileConfiguration
   [<CommonParameters>]
```

Description

This command returns a byte array (or "blob"), which can be passed to the Get-UserProfileDefinition and Set-UserProfileDefinition commands to inspect and modify a configuration set.

Use this command to bootstrap the configuration process in the case where no configuration policy objects are available in the Broker's database, or when you wish to create a fresh configuration that is separate from any that already exist.

No settings will be explicitly configured in the new configuration set. The properties of the configuration will be based on documented default values. Use Set-UserProfileDefinition to alter the state of any property within the configuration set.

Examples

EXAMPLE 1

The first command creates a fresh configuration set in its default state, and stores it in a Windows PowerShell variable. The second command interprets the new blob, and would output the individual properties of the default configuration.

```
$blob = New-UserProfileConfiguration

Get-UserProfileDefinition -ByteArray $blob
```

EXAMPLE 2

This command creates a fresh configuration set in its default state, and pipes the resulting byte array through to Get-UserProfileDefinition, which will interpret it and output its individual properties.

```
New-UserProfileConfiguration | Get-UserProfileDefinition
```


Parameters

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

The new configuration set, which can be piped directly into Get-UserProfileDefinition to receive the named properties.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

New-UserProfileShare

March 11, 2024

Creates a new Windows network share on a nominated computer, making it suitable for use as a profile store.

Syntax

```
New-UserProfileShare
   -ServerName <String>
   -ShareName <String>
   -PathOnServer <String>
```



```
    -UserGroups <String[]>
    -AdminGroups <String[]>
    [<CommonParameters>]
```

Description

Use this command as part of the procedure for creating a new profile store when configuring profile management for first use. A suitable folder is assumed to already exist locally on the nominated computer. If no such folder exists, then you will need to create it on the machine before running this command. This command automates the process of transforming that folder into a published network share with suitable access control for profile management.

When using this command to publish a share, you will need to supply information about the security groups of users who will be storing their profiles on the share. You can also optionally specify security groups who will be given an administrative level of access, should your organisation require this.

Examples

EXAMPLE 1

This command creates a new network share called "Profiles$"on the machine called "ProfileServer". It does this by sharing the folder "C:\ProfileStore", which will already be resident on ProfileServer. The store can be used by all domain users in the FABRIKAM domains. The store (including all of the user profiles that are subsequently stored within it) will also be accessible to the domain administrators. Following successful execution of this command, the network share whose UNC name is "\\ProfileServer\Profiles$"will exist. This makes it possible, for instance, to establish "\\ProfileServer\Profiles$\%USERNAME%.%USERDOMAIN"as a valid store path when configuring profile management.

```
C:PS> New-UserProfileShare -ServerName ProfileServer -ShareName
    Profiles$ -PathOnServer C:\ProfileStore -UserGroups @("FABRIKAM\
    Domain Users") -AdminGroups @("FABRIKAM\Administrators")
```

Parameters

-ServerName

This string specifies the network identity of the computer on which the share is being published.

Type: String

Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-ShareName

This specifies the desired name of the share itself. It is recommended (although not required) that share names end with the "$"character, so that they are excluded from casual enumerations of available shares.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-PathOnServer

This string specifies the absolute path of the local directory on the server computer, which will become the shared folder. This path must make sense from the point of view of the server, so it must not be a UNC path. A path might, for example, be of the form "C:\UserProfiles\FinanceDepartment".

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-UserGroups

This array of strings specifies the names of security groups whose members will be creating their profiles in the store. This array must contain at least one entry. Each string must be specified as a Windows object identifier, such as "FABRIKAM\Domain Users".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminGroups

This array of strings specifies the names of security groups that should be given an administrative level of access to the profile share. If no such access is required, the array can be supplied as empty. Each string must be specified as a Windows object identifier, such as "FABRIKAM\Administrators".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

Remove-UserProfileShare

March 11, 2024

Removes a nominated network share.

Syntax

```
Remove-UserProfileShare
      [-Path] <String>
      [<CommonParameters>]
```

Description

This command might be used as part of a final tear-down procedure for a profile store that is no longer needed.

This command will not delete any folders or data from the server computer, nor will it alter any access permissions on the folder. It will only restore the shared folder into an unshared state.

Examples

EXAMPLE 1

This command will unpublish the network share called "Profiles$"on the computer called "Profile-Server". The contents of the folder itself will not be affected. Any required deletion of profile store contents must be performed manually by an appropriate administrator of the folder.

```
C:PS> Remove-UserProfileShare -Path \\ProfileServer\Profiles$
```

Parameters

-Path

This string specifies the full UNC name of the network share to be removed. Since it is a UNC name, it will naturally specify both the name of the server computer and the name of the share to be removed. Consequently, no additional inputs are required.

Type:	String
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

Repair-UserProfileShare

March 11, 2024

Carries out one or more remedial actions on a profile store.

Syntax

```
Repair-UserProfileShare
      -Path <String>
      [-Mandatory]
      [-RedirectedFolders]
      -UserGroups <String[]>
      -AdminGroups <String[]>
      [<CommonParameters>]
```

```
Repair-UserProfileShare
      -Path <String>
      [-Mandatory]
      [-RedirectedFolders]
      -UserGroupsSids <String[]>
      -AdminGroupsSids <String[]>
      [<CommonParameters>]
```

```
Repair-UserProfileShare
      -Path <String>
      [-Mandatory]
      [-RedirectedFolders]
      [<CommonParameters>]
```


Description

Use this command in situations where the Test-UserProfileShare cmdlet reported one or more fixable faults with the profile store.

The inputs and outputs of this command are identical to those of %5BTest-UserProfileShare%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/UserProfileManager/Test-UserProfileShare.html). The only difference between the two cmdlets is that this cmdlet will attempt to fix the issues, where %5BTest-UserProfileShare%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/UserProfileManager/Test-UserProfileShare.html) will only report them.

Issues that cannot be fixed will be reported as test failures, just as they would with Test-UserProfileShare.

This cmdlet can only be used to fix issues that were claimed to be fixable in the output report from Test-UserProfileShare.

Examples

EXAMPLE 1

This command ensures that the path "\\ProfileServer\Profiles$\%USERAME%.%USERDOMAIN%"is suitable for use as a profile store. This path is syntactically valid, so the cmdlet will check for the existence of the share "\\ProfileServer\Profiles$". Provided that the share exists, it will also ensure that its permissions are configured such that all domain users would be permitted to create profile folders within the store, and that all domain administrators will be permitted to access the store for administrative purposes.

```
C:PS> Repair-UserProfileShare -Path \\ProfileServer\Profiles$\%USERNAME
    %.%USERDOMAIN% -UserGroups @("FABRIKAM\Domain Users") -AdminGroups @
    ("FABRIKAM\Administrators")
```

EXAMPLE 2

This command ensures that the path "\\ProfileServer\Profiles$\FinanceDept\FixedProfile"contains a valid Citrix mandatory profile. This path is syntactically valid, so the cmdlet will check that the share "\\ProfileServer\Profiles$"exists. If it exists, the cmdlet will go on to ensure that its permissions are configured such that all domain users would be permitted to read data from the profile, and that all domain administrators will be permitted to access the profile for administrative purposes.

Additional repair actions will be performed on the contents of the profile to ensure that there is nothing within the profile that might reveal details about the user who logged in to create it.


```
C:PS> Repair-UserProfileShare -Path \\ProfileServer\Profiles$\
    FinanceDept\FixedProfile -UserGroups @("FABRIKAM\Domain Users") -
    AdminGroups @("FABRIKAM\Administrators") -Mandatory
```

Parameters

-Path

This string contains the prospective profile store path to be examined, such as "\\ProfileServer\Profiles$\%USERNAME%.%USERDOMAIN%".

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-UserGroups

This array of strings specifies the names of security groups whose members will be creating their profiles in the store. This array must contain at least one entry. Each string must be specified as a Windows object identifier, such as "FABRIKAM\Domain Users".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminGroups

This array of strings specifies the names of security groups that should be given an administrative level of access to the profile share. If no such access is required, the array can be supplied as empty.

Each string must be specified as a Windows object identifier, such as "FABRIKAM\Administrators".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-UserGroupsSids

This array of strings specifies the SID of security groups whose members will be creating their profiles in the store. This array must contain at least one entry. Each string must be specified as security identifier string, such as "S-1-1-0".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminGroupsSids

This array of strings specifies the SID of security groups that should be given an administrative level of access to the profile share. If no such access is required, the array can be supplied as empty. Each string must be specified as security identifier string, such as "S-1-5-32-544".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False

Accept wildcard characters:	False

-Mandatory

This switch indicates whether the profile store is intended to house a mandatory profiles. Mandatory profile stores are subjected to additional checks.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RedirectedFolders

This switch indicates whether the path indicates the location of redirected folder directories.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

This cmdlet outputs a sequence of objects, each of which has the following properties:

Name - This property indicates the name of the test.

Status - This property indicates whether the repair action succeeded or failed.

Details - This property provides additional information about the test in the case where it has failed. The interpretation of this property depends on the name of the test.

CanFix - In the case where the test has failed, this boolean property indicates whether the problem can be fixed with the Repair-UserProfileShare cmdlet.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

Set-UserProfileDefinition

March 11, 2024

Applies one or more modifications to a profile management configuration set.

Syntax

```
Set-UserProfileDefinition
   [-ByteArray] <Byte[]>
   [-ServiceActive <Boolean>]
   [-ProcessedGroups <String[]>]
   [-ExcludedGroups <String[]>]
   [-ProcessAdmins <Boolean>]
   [-UserStorePath <String>]
   [-PSMidSessionWriteBack <Boolean>]
```



```
    [-OfflineSupport <Boolean>]
    [-DeleteCachedProfilesOnLogoff <Boolean>]
    [-ProfileDeleteDelay <Int32>]
    [-MigrateWindowsProfilesToUserStore <
       MigrateWindowsProfilesToUserStoreEnum>]
    [-LocalProfileConflictHandling <LocalProfileConflictHandlingEnum>]
    [-TemplateProfilePath <String>]
    [-TemplateProfileOverridesLocalProfile <Boolean>]
    [-TemplateProfileOverridesRoamingProfile <Boolean>]
    [-TemplateProfileIsMandatory <Boolean>]
    [-LoadRetries <Int32>]
    [-ProcessCookieFiles <Boolean>]
    [-DisableDynamicConfig <Boolean>]
    [-LogoffRatherThanTempProfile <Boolean>]
    [-DebugMode <Boolean>]
    [-LogLevel_Warnings <Boolean>]
    [-LogLevel_Information <Boolean>]
    [-LogLevel_FileSystemNotification <Boolean>]
    [-LogLevel_FileSystemActions <Boolean>]
    [-LogLevel_RegistryActions <Boolean>]
    [-LogLevel_RegistryDifference <Boolean>]
    [-LogLevel_ActiveDirectoryActions <Boolean>]
    [-LogLevel_PolicyUserLogon <Boolean>]
    [-LogLevel_Logon <Boolean>]
    [-LogLevel_Logoff <Boolean>]
    [-LogLevel_UserName <Boolean>]
    [-MaxLogSize <Int32>]
    [-DebugFilePath <String>]
    [-ExclusionList <String[]>]
    [-IncludeListRegistry <String[]>]
    [-ExclusionListSyncFiles <String[]>]
    [-ExclusionListSyncDir <String[]>]
    [-SyncDirList <String[]>]
    [-SyncFileList <String[]>]
    [-MirrorFoldersList <String[]>]
    [-PSEnabled <Boolean>]
    [-PSAlwaysCache_Enabled <Boolean>]
    [-PSAlwaysCache <Int32>]
    [-PSPendingLockTimeout <Int32>]
    [-PSUserGroups <String[]>]
    [-CPEnable <Boolean>]
    [-CPUserGroups <String[]>]
    [-CPSchemaPathData <String>]
    [-CPPathData <String>]
    [-CPMigrationFromBaseProfileToCPStore <Boolean>]
    [-FRAdminAccess <Boolean>]
    [-FRIncDomainName <Boolean>]
    [-FRAppDataEnabled <FRAppDataEnum>]
    [-FRAppDataPath <String>]
    [-FRDesktopEnabled <FRDesktopEnum>]
    [-FRDesktopPath <String>]
    [-FRStartMenuEnabled <FRStartMenuEnum>]
    [-FRStartMenuPath <String>]
```



```
61      [-FRDocumentsEnabled <FRDocumentsEnum>]
62      [-FRDocumentsPath <String>]
63      [-FRPicturesEnabled <FRPicturesEnum>]
64      [-FRPicturesPath <String>]
65      [-FRMusicEnabled <FRMusicEnum>]
66      [-FRMusicPath <String>]
67      [-FRVideosEnabled <FRVideosEnum>]
68      [-FRVideosPath <String>]
69      [-FRFavoritesEnabled <FRFavoritesEnum>]
70      [-FRFavoritesPath <String>]
71      [-FRContactsEnabled <FRContactsEnum>]
72      [-FRContactsPath <String>]
73      [-FRDownloadsEnabled <FRDownloadsEnum>]
74      [-FRDownloadsPath <String>]
75      [-FRLinksEnabled <FRLinksEnum>]
76      [-FRLinksPath <String>]
77      [-FRSearchesEnabled <FRSearchesEnum>]
78      [-FRSearchesPath <String>]
79      [-FRSavedGamesEnabled <FRSavedGamesEnum>]
80      [-FRSavedGamesPath <String>]
81      [<CommonParameters>]
```

Description

Use this cmdlet to edit the configurations for profile management. The cmdlet defines one input parameter for each settable property. Any number of these settings can be modified in a single call to the cmdlet. The cmdlet also requires the low-level byte array blob containing the original configuration set to be modified. It will return an updated byte array with the settings applied. The resulting byte array can then be stored back at the Broker.

Examples

EXAMPLE 1

The first command creates a new user profile configuration in its default state, and stores the byte array in a Windows PowerShell variable. The second command then edits this configuration by setting the ServiceActive property to TRUE (which has the effect of enabling the profile management service), and by setting the path to the profile store. All other settings remain unaffected, and hence in their default state.

```
1  C:PS> $blob = New-UserProfileConfiguration
2
3  C:PS> $updatedBlob = Set-UserProfileDefinition -ByteArray $blob -
       ServiceActive $true -UserStorePath \\ProfileServer\Profiles$\%
       USERNAME%.%USERDOMAIN
```


Parameters

-ByteArray

Specifies the original configuration set that is being modified by this invocation of the command.

Type:	Byte[]
Position:	1
Default value:	None
Required:	True
Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-ServiceActive

Applies to: both file-based and container-based profile solutions

By default, to facilitate deployment, Profile management does not process logons or logoffs.

Turn on processing by enabling this setting.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, Profile management does not process Windows user profiles in any way.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProcessedGroups

Applies to: both file-based and container-based profile solutions

Both computer local groups and domain groups (local, global and universal) can be used. Domain groups should be specified in the format: <DOMAIN NAME>\<GROUP NAME>.

If this setting is configured here, Profile management processes only members of these user groups.

If this setting is disabled, Profile management processes all users.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, members of all user groups are processed.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExcludedGroups

Applies to: both file-based and container-based profile solutions

You can use computer local groups and domain groups (local, global, and universal) to prevent particular user profiles from being processed. Specify domain groups in the form <DOMAIN NAME>\<GROUP NAME>.

If this setting is configured here, Profile management excludes members of these user groups.

If this setting is disabled, Profile management does not exclude any users.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no members of any groups are excluded.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProcessAdmins

Applies to: both file-based and container-based profile solutions

Specifies whether logons of members of the local group "Administrators"are processed by Profile management.

If this setting is disabled, logons by local administrators are not processed by Profile management.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, administrators will not be processed.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-UserStorePath

Applies to: both file-based and container-based profile solutions

Sets the path to the directory in which the user settings (registry changes and synchronized files) are saved (user store).

The path can be an absolute UNC path or a path relative to the home directory.

In both cases, the following types of variables can be used: system environment variables enclosed in percent signs and attributes of the Active Directory user object enclosed in hashes.

Examples: The folder Windows\%ProfileVer% stores the user settings in the subfolder called Windows\W2k3 of the user store (if %ProfileVer% is a system environment variable resolving to W2k3).

\\server\share#SAMAccountName# stores the user settings to the UNC path \\server\share\JohnSmith (if #SAMAccountName# resolves to JohnSmith for the current user).

User environment variables cannot be used, except for %username% and %userdomain%.

If this setting is disabled, the user settings are saved in the Windows subdirectory of the home directory.

If this setting is not configured here, the setting from the .ini file is used.

If this setting is not configured here or in the .ini file, the Windows directory on the home drive is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PSMidSessionWriteBack

Applies to: file-based profile solution only

With this setting, files and folders (but not Registry entries) that are modified can be synchronized to the user store in the middle of a session, before logoff.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, active write back is disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-OfflineSupport

Applies to: file-based profile solution only

Enables the offline profiles feature. This is intended for computers that are commonly removed from networks, typically laptops or mobile devices not servers or desktops.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, offline profile support is disabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DeleteCachedProfilesOnLogoff

Applies to: file-based profile solution only

Specifies whether locally cached profiles are deleted after logoff.

If this setting is enabled, a user's local profile cache is deleted after they have logged off. This is recommended for terminal servers.

If this setting is disabled cached profiles are not deleted.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, cached profiles are not deleted.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProfileDeleteDelay

Applies to: file-based profile solution only

Works only if 'Delete locally cached profiles on logoff'is enabled. Sets an optional extension to the delay before locally cached profiles are deleted at logoff. A value of 0 deletes the profiles immediately during logoff processing. Checks take place every minute, so a value of 60 ensures that profiles are deleted between one and two minutes after users have logged off (depending on when the last check

took place). Extending the delay is useful if you know that a process keeps files or the user registry hive open during logoff. With large profiles, this can also speed up logoff.

If this policy is not configured here, the value from the .ini file is used.

If this policy is not configured here or in the .ini file, profiles are deleted immediately.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MigrateWindowsProfilesToUserStore

Applies to: file-based profile solution only

Profile management can migrate existing profiles "on the fly"during logon if the user has no profile in the user store.

The following event takes place during logon: if an existing Windows profile is found and the user does not yet have a Citrix user profile in the user store, the Windows profile is migrated (copied) to the user store on the fly. After this process, the user store profile is used by Profile management in the current and any other session configured with the path to the same user store.

If this setting is enabled, profile migration can be activated for roaming and local profiles (the default), roaming profiles only, local profiles only, or profile migration can be disabled altogether.

If profile migration is disabled and no Citrix user profile exists in the user store, the existing Windows mechanism for creating new profiles is used as in a setup without Profile management.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all types of existing profiles are migrated.

Type:	MigrateWindowsProfilesToUserStoreEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LocalProfileConflictHandling

Applies to: file-based profile solution only

This setting configures what Profile management does if both a profile in the user store and a local Windows user profile (not a Citrix user profile) exist.

If this setting is disabled or set to the default value of "Use local profile", Profile management uses the local profile, but does not change it in any way.

If this setting is set to "Delete local profile", Profile management deletes the local Windows user profile, and then imports the Citrix user profile from the user store.

If this setting is set to "Rename local profile", Profile management renames the local Windows user profile (in order to back it up) and then imports the profile from the user store.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, existing local profiles are used.

Type:	LocalProfileConflictHandlingEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TemplateProfilePath

Applies to: both file-based and container-based profile solutions

By default, new user profiles are created from the default user profile on the computer where a user first logs on. Profile management can alternatively use a centrally stored template when creating new user profiles. Template profiles are identical to normal profiles in that they reside in any file share on the network. Use UNC notation to specifying paths to templates. Users need read access to a template profile.

If this setting is disabled, templates are not used.

If this setting is enabled, Profile management uses the template instead of the local default profile when creating new user profiles.

If a user has no Citrix user profile, but a local Windows user profile exists, by default the local profile is used (and migrated to the user store, if this is not disabled). This can be changed by enabling the setting "Template profile overrides local profile".

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TemplateProfileOverridesLocalProfile

Applies to: both file-based and container-based profile solutions

By default, new user profiles are created from the default user profile on the computer where a user first logs on. Profile management can alternatively use a centrally stored template when creating new user profiles. Template profiles are identical to normal profiles in that they reside in any file share on the network. Use UNC notation to specifying paths to templates. Users need read access to a template profile.

If this setting is disabled, templates are not used.

If this setting is enabled, Profile management uses the template instead of the local default profile when creating new user profiles.

If a user has no Citrix user profile, but a local Windows user profile exists, by default the local profile is used (and migrated to the user store, if this is not disabled). This can be changed by enabling the setting "Template profile overrides local profile".

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TemplateProfileOverridesRoamingProfile

Applies to: both file-based and container-based profile solutions

By default, new user profiles are created from the default user profile on the computer where a user first logs on. Profile management can alternatively use a centrally stored template when creating new user profiles. Template profiles are identical to normal profiles in that they reside in any file share on the network. Use UNC notation to specifying paths to templates. Users need read access to a template profile.

If this setting is disabled, templates are not used.

If this setting is enabled, Profile management uses the template instead of the local default profile when creating new user profiles.

If a user has no Citrix user profile, but a local Windows user profile exists, by default the local profile is used (and migrated to the user store, if this is not disabled). This can be changed by enabling the setting "Template profile overrides local profile".

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-TemplateProfileIsMandatory

Applies to: both file-based and container-based profile solutions

By default, new user profiles are created from the default user profile on the computer where a user first logs on. Profile management can alternatively use a centrally stored template when creating new user profiles. Template profiles are identical to normal profiles in that they reside in any file share on the network. Use UNC notation to specifying paths to templates. Users need read access to a template profile.

If this setting is disabled, templates are not used.

If this setting is enabled, Profile management uses the template instead of the local default profile when creating new user profiles.

If a user has no Citrix user profile, but a local Windows user profile exists, by default the local profile is used (and migrated to the user store, if this is not disabled). This can be changed by enabling the setting "Template profile overrides local profile".

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no template is used.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoadRetries

Applies to: both file-based and container-based profile solutions

Sets the number of retries when accessing locked files.

If this setting is disabled the default value of five retries is used.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default value of five retries is used.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-ProcessCookieFiles

Applies to: file-based profile solution only

Some deployments leave extra Internet cookies that are not referenced by the file index.dat. The extra cookies left in the file system after sustained browsing can lead to profile bloat. Enable this setting to force processing of index.dat and remove the extra cookies. The setting might slightly increase logoff time. Enable this setting only when you experience the issue.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no processing of index.dat takes place.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DisableDynamicConfig

Applies to: both file-based and container-based profile solutions

Profile management 5.x examines its environment and configures itself accordingly. To disable this for troubleshooting or to retain the settings you used in an earlier version, enable this setting.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, automatic configuration is turned on so Profile management settings might change if the environment changes.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogoffRatherThanTempProfile

Applies to: both file-based and container-based profile solutions

If a problem is encountered when the user logs on (for example, if the user store is unavailable) a temporary profile is provided.

Alternatively, enabling this setting displays an error message and logs the user off.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, a temporary profile is provided.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DebugMode

Applies to: both file-based and container-based profile solutions

Activation of this setting enables debug mode (verbose logging). In debug mode, extensive status information is logged in the log files in "%SystemRoot%\System32\Logfiles\UserProfileManager"or in the location specified by the "Path to log file"policy setting.

If this setting is disabled only errors are logged.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, only errors are logged.

Type:	Boolean

Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_Warnings

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_Information

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_FileSystemNotification

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_FileSystemActions

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_RegistryActions

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_RegistryDifference

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_ActiveDirectoryActions

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_PolicyUserLogon

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-LogLevel_Logon

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_Logoff

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LogLevel_UserName

Applies to: both file-based and container-based profile solutions

Detailed log settings.

Define events or actions that Profile management logs in depth.

If this setting is not configured here, Profile management uses the settings from the .ini file.

If this setting is not configured here or in the .ini file, errors and general information are logged.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MaxLogSize

Applies to: both file-based and container-based profile solutions

Sets the maximum size of the log file in bytes. If the log file grows beyond this size an existing backup of the file (.bak) is deleted, the log file is renamed to .bak, and a new log file is created.

The log file is created in "%SystemRoot%\System32\Logfiles\UserProfileManager"or in the location specified by the "Path to log file"policy setting.

If this setting is disabled, the default value of 10 MB is used.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default of 10 MB is used.

Type:	Int32
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-DebugFilePath

Applies to: both file-based and container-based profile solutions

Sets an alternative path to which the log files are saved.

The path can point to a local drive or a remote, network-based one (a UNC path). Remote paths can be useful in large, distributed environments but they can create significant network traffic, which may be inappropriate for log files. For provisioned, virtual machines with a persistent hard drive, set a local path to that drive. This ensures log files are preserved when the machine restarts. For virtual machines without a persistent hard drive, setting a UNC path allows you to retain the log files but the system account for the machines must have write access to the UNC share. Use a local path for any laptops managed by the offline profiles feature.

Examples: D:\LogFiles\ProfileManagement. \\servername\LogFiles\ProfileManagement

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default location "%SystemRoot%\System32\Logfiles\UserP
is used.

If a UNC path is used for log files then Citrix recommends that an appropriate access control list is applied to the log file folder to ensure that only authorized user or computer accounts can access the stored files.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExclusionList

Applies to: file-based profile solution only

List of registry keys in the HKCU hive that are ignored during logoff.

Example: Software\Policies.

If this setting is disabled, no registry keys are excluded.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no registry keys are excluded.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-IncludeListRegistry

Applies to: file-based profile solution only

List of registry keys in the HKCU hive that are processed during logoff. Example: Software\Adobe.

If this setting is enabled, only keys on this list are processed.

If this setting is disabled, the complete HKCU hive is processed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all of HKCU is processed.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExclusionListSyncFiles

Applies to: file-based profile solution only

List of files that are ignored during synchronization.

File names should be specified as paths relative to the user profile. Wildcards are allowed. Wildcards in file names are applied recursively while wildcards in folder names are not.

Note: As of Profile Management 7.15, you can use the vertical bar '|'for applying a policy to only the current folder without propagating it to the subfolders.

Examples: Desktop\Desktop.ini ignores the Desktop.ini file in the Desktop folder.

AppData*.tmp ignores all files with the .tmp extension in the AppData folder and its subfolders.

AppData*.tmp| ignores all files with the .tmp extension in the AppData folder.

Downloads*\a.txt ignores a.txt in any immediate subfolder of the Downloads folder. Note that wildcards in folder names are not applied recursively.

If this setting is disabled, no files are excluded.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no files are excluded.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ExclusionListSyncDir

Applies to: file-based profile solution only

List of directories that are ignored during synchronization.

Folder names should be specified as paths relative to the user profile.

Wildcards are supported but they are not applied recursively.

Examples: Desktop ignores the Desktop folder in the user profile.

Downloads* ignores all immediate subfolders of the Downloads folder.

If this setting is disabled, no folders are excluded.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no folders are excluded.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SyncDirList

Applies to: file-based profile solution only

Profile management synchronizes each user's entire profile between the system it is installed on and the user store.

It allows you to include subfolders of excluded folders.

Paths on this list should be relative to the user profile.

Wildcards are supported but they are not applied recursively.

Examples: Desktop\exclude\include specifies the include subfolder of the Desktop\exclude folder.

Desktop\exclude* specifies all immediate subfolders of the Desktop\exclude folder.

Disabling this setting has the same effect as enabling it and configuring an empty list.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, only non-excluded folders in the user profile are synchronized.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SyncFileList

Applies to: file-based profile solution only

Profile management synchronizes each user's entire profile between the system it is installed on and the user store.

This setting allows for the inclusion of files below excluded folders.

Paths on this list should be relative to the user profile. Wildcards are allowed. Wildcards in file names are applied recursively while wildcards in folder names are not.

Note: As of Profile Management 7.15, you can use the vertical bar '|'for applying a policy to only the current folder without propagating it to the subfolders.

Examples: AppData\Local\Microsoft\Office\Access.qat specifies a file below a folder excluded in the default configuration.

AppData\Local\MyApp*.cfg specifies all files with the extension .cfg in the profile folder AppData\Local\MyApp and its subfolders.

AppData\Local\MyApp*.cfg| specifies all files with the extension .cfg in the profile folder AppData\Local\MyApp.

Downloads*\b.txt specifies b.txt in any immediate subfolder of the Downloads folder. Note that wildcards in folder names are not applied recursively.

Disabling this setting has the same effect as enabling it and configuring an empty list.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, only non-excluded files in the user profile are synchronized.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MirrorFoldersList

Applies to: file-based profile solution only

Profile management can mirror a folder relative to the profile's root folder. Use this setting for files whose contents index data and where separate instances of the data are likely to exist. For example, you can mirror the Internet Explorer cookies folder so that index.dat is synchronized with the cookies that it indexes. Be aware that, in these situations the "last write wins"so files in mirrored folders that have been modified in more than one session are overwritten by the last update, resulting in loss of profile changes.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PSEnabled

Applies to: file-based profile solution only

With profile streaming, users'profiles are synchronized on the local computer only when they are needed. Registry entries are cached immediately, but files and folders are only cached when accessed by users.

Type:	Boolean
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PSAlwaysCache_Enabled

Applies to: file-based profile solution only

Optionally, to enhance the user experience, use this setting with the Profile streaming setting, which imposes a lower limit on the size of files that are streamed. Any files this size or larger are cached as soon as possible after logon. To use the cache entire profile feature, set this limit to zero (which caches all of the profile contents).

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PSAlwaysCache

Applies to: file-based profile solution only

Optionally, to enhance the user experience, use this setting with the Profile streaming setting, which imposes a lower limit on the size of files that are streamed. Any files this size or larger are cached as soon as possible after logon. To use the cache entire profile feature, set this limit to zero (which caches all of the profile contents).

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-PSPendingLockTimeout

Applies to: file-based profile solution only

You can set a timeout period that frees up users' files so they are written back to the user store from the pending area in the event that the user store remains locked when a server becomes unresponsive (for example, when it goes down). Use this setting to prevent bloat in the pending area and to ensure the user store always contains the most up-to-date files.

Type:	Int32
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PSUserGroups

Applies to: file-based profile solution only

Enter one or more Windows user groups.

If this setting is enabled, only the profiles of those groups' members are streamed. If this setting is disabled, all user groups are processed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all users are processed.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-CPEnable

Applies to: both file-based and container-based profile solutions

By default, to facilitate deployment, cross-platform settings is disabled.

Turn on processing by enabling this setting.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no cross-platform settings will be applied.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CPUserGroups

Applies to: both file-based and container-based profile solutions

Enter one or more Windows user groups.

If this setting is configured, the cross-platform settings feature of Profile management processes only members of these user groups. If this setting is disabled, all user groups are processed.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, all user groups are processed.

Note: Use Enter to separate multiple entries.

Type:	String[]
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-CPSchemaPathData

Applies to: both file-based and container-based profile solutions

Specify a folder where the definition files are located.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, no cross-platform settings will be applied.

Note:

- The folder can be local or it can reside on an SMB file share.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CPPathData

Applies to: both file-based and container-based profile solutions

Sets the path to the directory in which users' cross-platform settings are saved (cross-platform settings store).

The path can be an absolute UNC path or a path relative to the home directory.

In both cases, the following types of variables can be used: system environment variables enclosed in percent signs and attributes of the Active Directory user object enclosed in hashes.

User environment variables cannot be used, except for %username% and %userdomain%.

If this setting is disabled, the path Windows\PM_CP is used.

If this setting is not configured here, the value from the .ini file is used.

If this setting is not configured here or in the .ini file, the default value Windows\PM_CP is used.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-CPMigrationFromBaseProfileToCPStore

Applies to: both file-based and container-based profile solutions

Default: Disabled

Definition files contain a set of definitions that configure an application. If the cross-platform settings store contains a definition with no data, or the cached data in the single-platform profile is newer than the definition's data in the store, Profile management only migrates the data from the single-platform profile to the store if you enable this setting.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRAdminAccess

Applies to: file-based profile solution only

By default, users are granted exclusive access to the contents of their redirected folders.

Enabling this option allows administrator access to the contents of the users redirected folders.

Type:	Boolean
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRIncDomainName

Applies to: file-based profile solution only

Enabling this option will include the %userdomain% environment variable as part of the UNC path.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRAppDataEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the AppData(Roaming) folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.

2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRAppDataEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRAppDataPath

Applies to: file-based profile solution only

Lets you specify how to redirect the AppData(Roaming) folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FRDesktopEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Desktop folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRDesktopEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRDesktopPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Desktop folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRStartMenuEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Start Menu folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRStartMenuEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRStartMenuPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Start Menu folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRDocumentsEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Documents folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRDocumentsEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRDocumentsPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Documents folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRPicturesEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Pictures folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected

folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRPicturesEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRPicturesPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Pictures folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.

2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRMusicEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Music folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRMusicEnum
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FRMusicPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Music folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRVideosEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Videos folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRVideosEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRVideosPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Videos folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRFavoritesEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Favorites folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRFavoritesEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRFavoritesPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Favorites folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRContactsEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Contacts folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRContactsEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRContactsPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Contacts folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected

folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRDownloadsEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Downloads folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.

2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRDownloadsEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRDownloadsPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Downloads folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-FRLinksEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Links folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRLinksEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRLinksPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Links folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRSearchesEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Searches folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRSearchesEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRSearchesPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Searches folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRSavedGamesEnabled

Applies to: file-based profile solution only

Lets you specify how to redirect the Saved Games folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	FRSavedGamesEnum
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-FRSavedGamesPath

Applies to: file-based profile solution only

Lets you specify how to redirect the Saved Games folder. To do so, select Enabled and then type the redirected path. Caution: Potential data loss might occur. See below for details. You might want to modify the path after the policy takes effect. However, consider potential data loss before you do so. The data contained in the redirected folder might be deleted if the modified path points to the same location as the previous path. For example, suppose you specify the Contacts path as path1. Later, you change path1 to path2. If path1 and path2 point to the same location, all data contained in the redirected folder is deleted after the policy takes effect. To avoid potential data loss, complete the following steps:

1. Apply Microsoft policy to machines where Profile Management is running through Active Directory Group Policy Objects. To do so, open the Group Policy Management Console, navigate to Computer Configuration > Administrative Templates > Windows Components > File Explorer, and then enable Verify old and new Folder Redirection targets point to the same share before redirecting.
2. If applicable, apply hotfixes to machines where Profile Management is running. For details, see https://support.microsoft.com/en-us/help/977229 and https://support.microsoft.com/en-us/help/2799904.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

Byte[]

The new configuration set, with all of the given modifications applied.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

Test-UserProfileShare

March 11, 2024

Tests a prospective user profile store path for its suitability for use with Citrix Profile Management.

Syntax

```
Test-UserProfileShare
    -Path <String>
    [-Mandatory]
    [-RedirectedFolders]
    -UserGroups <String[]>
    -AdminGroups <String[]>
    [<CommonParameters>]
```

```
Test-UserProfileShare
    [-PathSyntaxOnly]
    -Path <String>
    [-Mandatory]
    [-RedirectedFolders]
    [<CommonParameters>]
```

```
Test-UserProfileShare
    -Path <String>
    [-Mandatory]
```



```
    [-RedirectedFolders]
    -UserGroupsSids <String[]>
    -AdminGroupsSids <String[]>
    [<CommonParameters>]
```

Description

This command can be used as a validation step prior to configuring the profile store path in a production environment, in order to gain confidence that the profile manager and the store will function together correctly.

This command runs a sequence of tests on the nominated path, and reports the results via the output pipeline. The tests are organised into two distinct phases. The first phase examines the syntax of the supplied store path string. The second phase examines the profile store itself, checking that the required network shares exist, and that their permissions are appropriately configured according to best practices. If any faults are discovered with the syntax of the path string, these will be reported via the output pipeline, and the command will halt early without attempting to examine the profile store. The profile store will only be examined if the path is syntactically correct.

This command can be used in combination with the Repair-UserProfileShare cmdlet, which is able to automatically correct some of the problems that are found during inspection of the profile store. Problems with the syntax of the path can never be repaired. Syntactic issues must be fixed by the administrator re-typing the path and testing the store again.

Examples

EXAMPLE 1

This command tests that the path "\\ProfileServer\Profiles$\%USERAME%.%USERDOMAIN%"is suitable for use as a profile store. This path is syntactically valid, so the cmdlet will check for the existence of the share "\\ProfileServer\Profiles$". Provided that the share exists, it will also check that its permissions are configured such that all domain users would be permitted to create profile folders within the store, and that all domain administrators will be permitted to access the store for administrative purposes.

```
C:PS> Test-UserProfileShare -Path \\ProfileServer\Profiles$\%USERNAME
    %.%USERDOMAIN% -UserGroups @("FABRIKAM\Domain Users") -AdminGroups @
    ("FABRIKAM\Administrators")
```


EXAMPLE 2

This command tests that the path "\\ProfileServer\Profiles$\FinanceDept\FixedProfile"contains a valid Citrix mandatory profile. This path is syntactically valid, so the cmdlet will check that the share "\\ProfileServer\Profiles$"exists. If it exists, the cmdlet will go on to check that its permissions are configured such that all domain users would be permitted to read data from the profile, and that all domain administrators will be permitted to access the profile for administrative purposes.

Additional checks will be performed on the contents of the profile to ensure that there is nothing within the profile that might reveal details about the user who logged in to create it.

```
C:PS> Test-UserProfileShare -Path \\ProfileServer\Profiles$\FinanceDept
    \FixedProfile -UserGroups @("FABRIKAM\Domain Users") -AdminGroups @(
    "FABRIKAM\Administrators") -Mandatory
```

Parameters

-PathSyntaxOnly

This switch indicates the the command should only check the syntax of the user store path.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Path

This string contains the prospective profile store path to be examined, such as "\\Profile-Server\Profiles$\%USERNAME%.%USERDOMAIN%".

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	True (ByValue)
Accept wildcard characters:	False

-UserGroups

This array of strings specifies the names of security groups whose members will be creating their profiles in the store. This array must contain at least one entry. Each string must be specified as a Windows object identifier, such as "FABRIKAM\Domain Users".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminGroups

This array of strings specifies the names of security groups that should be given an administrative level of access to the profile share. If no such access is required, the array can be supplied as empty. Each string must be specified as a Windows object identifier, such as "FABRIKAM\Administrators".

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-UserGroupsSids

This array of strings specifies the SID of security groups whose members will be creating their profiles in the store. This array must contain at least one entry. Each string must be specified as security

identifier string, such as “S-1-1-0”.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminGroupsSids

This array of strings specifies the SID of security groups that should be given an administrative level of access to the profile share. If no such access is required, the array can be supplied as empty. Each string must be specified as security identifier string, such as “S-1-5-32-544”.

Type:	String[]
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-Mandatory

This switch indicates whether the profile store is intended to house a mandatory profiles. Mandatory profile stores are subjected to additional checks.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-RedirectedFolders

This switch indicates whether the path indicates the location of redirected folder directories.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You can't pipe objects to this cmdlet.

Outputs

PSObject

This cmdlet outputs a sequence of objects, each of which has the following properties:

Name - This property indicates the name of the test.

Status - This property indicates whether the test has passed or failed.

Details - This property provides additional information about the test in the case where it has failed. The interpretation of this property depends on the name of the test.

CanFix - In the case where the test has failed, this boolean property indicates whether the problem can be fixed with the Repair-UserProfileShare cmdlet.

Related Links

- about_UpmSnapInUserProfileManagerSnapin

about_XD_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or

descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order. For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like ‘‘High*’’-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match. Separate parameters are combined with an implicit -and operator. Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company ‘‘citrix’’-Product ‘[X]EN*’
Get-<Noun> -Product ‘‘Xen*’’-Company ‘‘CITRIX’’
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:

Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:


```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
Get-<Noun> -Filter { StartTime -ge ‘-0:0:30’} # 30 seconds ago
```

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

```
Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like “Fred*”}
Get-<Noun> -Filter { Users -contains “Fred*”}
```


You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{
```



```
6
7    $sessions = @(Get-BrokerSession -Filter {
8    Uid -gt $lastUid  }
9    -MaxRecordCount 1000 -Sortby 'Uid')
```

```
if ($sessions.Length -eq 0)
{
break;
}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}
```

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

about_XD_Filtering

March 11, 2024

Topic

XenDesktop - Advanced Dataset Filtering

Short Description

Describes the common filtering options for XenDesktop cmdlets.

Long Description

Some cmdlets operate on large quantities of data and, to reduce the overhead of sending all of that data over the network, many of the Get- cmdlets support server-side filtering of the results.

The conventional way of filtering results in PowerShell is to pipeline them into Where-Object, Select-Object, and Sort-Object, for example:

Get-<Noun> | Where { $_.Size = 'Small'} | Sort 'Date'| Select -First 10

However, for most XenDesktop cmdlets the data is stored remotely and it would be slow and inefficient to retrieve large amounts of data over the network and then discard most of it. Instead, many of the Get- cmdlets provide filtering parameters that allow results to be processed on the server, returning only the required results.

You can filter results by most object properties using parameters derived from the property name. You can also sort results or limit them to a specified number of records:

Get-<Noun> -Size 'Small'-SortBy 'Date'-MaxRecordCount 10

You can express more complex filter conditions using a syntax and set of operators very similar to those used by PowerShell expressions.

Those cmdlets that support filtering have the following common parameters:

-MaxRecordCount <int>

Specifies the maximum number of results to return.
For example, to return only the first nine results use:

Get-<Noun> -MaxRecordCount 9

If not specified, only the first 250 records are returned, and if more are available, a warning is produced:

WARNING: Only first 250 records returned. Use -MaxRecordCount to retrieve more.

You can suppress this warning by using -WarningAction or by specifying a value for -MaxRecordCount.

Based on the use case, you can filter results or separate results into multiple pages. You may also use both approaches together.

-ReturnTotalRecordCount [<SwitchParameter>]

When specified, this causes the cmdlet to output an error record containing the number of records available. This error record is additional information and does not affect the objects written to the output pipeline. For example:

```
# ReturnTotalRecord example
Get-<Noun> -MaxRecordCount 9 -ReturnTotalRecordCount
```



```
....
Get-<Noun> : Returned 9 of 10 items
At line:1 char:18
+ Get-<Noun> <<<<  -MaxRecordCount 9 -ReturnTotalRecordCount
+ CategoryInfo          : OperationStopped: (:) [Get-<Noun>],
   PartialDataException
+ FullyQualifiedErrorId : PartialData,Citrix.<SDKName>.SDK.Get<Noun>
```

The count can be accessed using the TotalAvailableResultCount property:

$count = $error[0].TotalAvailableResultCount

-Skip <int>

Skips the specified number of records before returning results.
Also reduces the count returned by -ReturnTotalRecordCount.

-SortBy <string>

Sorts the results by the specified list of properties. The list is a set of property names separated by commas, semi-colons, or spaces. Optionally, prefix each name with a + or - to indicate ascending or descending order, respectively. Ascending order is assumed if no prefix is present.

Sorting occurs before -MaxRecordCount and -Skip parameters are applied. For example, to sort by Name and then by Count (largest first) use:

-SortBy ‘Name,-Count’

By default, sorting by an enumeration property uses the numeric value of the elements. You can specify a different sort order by qualifying the name with an ordered list of elements or their numeric values, or <null> to indicate the placement of null values.
Elements not mentioned are placed at the end in their numeric order.
For example, to sort by two different enums and then by the object id:

-SortBy ‘MyState(StateC,<null>,StateA,StateB),Another(0,3,2,1),Id’

-Filter <String>

This parameter lets you specify advanced filter expressions, and supports combination of conditions with -and and -or, and grouping with braces. For example:

Get-<Noun> -Filter ‘Name -like “High*”-or (Priority -eq 1 -and Severity -ge 2)’

The syntax is close enough to PowerShell syntax that you can use script blocks in most cases. This can be easier to read as it reduces

quoting:

Get-<Noun> -Filter { Count -ne $null }

The full -Filter syntax is provided below.

Examples

Filtering by strings performs a case-insensitive wildcard match.
Separate parameters are combined with an implicit -and operator.
Normal PowerShell quoting rules apply, so you can use single or double quotes, and omit the quotes altogether for many strings. The order of parameters does not make any difference. The following are equivalent:

Get-<Noun> -Company Citrix -Product Xen*
Get-<Noun> -Company “citrix”-Product ‘[X]EN*’
Get-<Noun> -Product “Xen*”-Company “CITRIX”
Get-<Noun> -Filter { Company -eq ‘Citrix’-and Product -like ‘Xen*’}

See about_Quoting_Rules and about_Wildcards for details about PowerShell handling of quotes and wildcards.

To avoid wildcard matching or include quote characters, you can escape the wildcards using the normal PowerShell escape mechanisms (see about_Escape_Characters), or switch to a filter expression and the -eq operator:

```
# Escape examples
Get-<Noun> -Company "Abc[*]"                # Matches Abc*
Get-<Noun> -Company "Abc`*"                 # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "Abc*"  }
     # Matches Abc*
Get-<Noun> -Filter {
  Company -eq "A`"B`'C"  }
  # Matches A"B'C
```

Simple filtering by numbers, booleans, and TimeSpans perform direct equality comparisons, although if the value is nullable you can also search for null values. Here are some examples:

```
# Simple filtering examples
Get-<Noun> -Uid 123
Get-<Noun> -Enabled $true
Get-<Noun> -Duration 1:30:40
Get-<Noun> -NullableProperty $null
```

More comparisons are possible using advanced filtering with -Filter:


```
Get-<Noun> -Filter ‘Capacity -ge 10gb’
Get-<Noun> -Filter ‘Age -ge 20 -and Age -lt 40’
Get-<Noun> -Filter ‘VolumeLevel -like “[123]”’
Get-<Noun> -Filter ‘Enabled -ne $false’
Get-<Noun> -Filter ‘NullableProperty -ne $null’
```

You can check boolean values without an explicit comparison operator, and you can also combine them with -not:

```
Get-<Noun> -Filter ‘Enabled’# Equivalent to ‘Enabled -eq $true’
Get-<Noun> -Filter ‘-not Enabled’# Equivalent to ‘Enabled -eq $false’
```

See about_Comparison_Operators for an explanation of the operators, but note that only a subset of PowerShell operators are supported (-eq, -ne, -gt, -ge, -lt, -le, -like, -notlike, -in, -notin, -contains, -notcontains).

Enumeration values can either be specified using typed values or the string name of the enumeration value:

```
Get-<Noun> -Shape [Shapes]::Square
Get-<Noun> -Shape Circle
```

With filter expressions, typed values can be specified with simple variables or quoted strings. They also support enumerations with wildcards:

```
$s = [Shapes]::Square
Get-<Noun> -Filter { Shape -eq $s -or Shape -eq “Circle”}
Get-<Noun> -Filter { Shape -like ‘C*’}
```

By their nature, floating point values, DateTime values, and TimeSpan values are best suited to relative comparisons rather than just equality. DateTime strings are converted using the locale and time zone of the user device, but you can use ISO8601 format strings (YYYY-MM-DDThh:mm:ss.sTZD) to avoid ambiguity. You can also use standard PowerShell syntax to create these values:

```
Get-<Noun> -Filter { StartTime -ge “2010-08-23T12:30:00.0Z”}
$d = [DateTime]“2010-08-23T12:30:00.0Z”
Get-<Noun> -Filter { StartTime -ge $d }
$d = (Get-Date).AddDays(-1)
Get-<Noun> -Filter { StartTime -ge $d }
```

Relative times are quite common and, when using filter expressions, you can also specify DateTime values using a relative format:

```
Get-<Noun> -Filter { StartTime -ge ‘-2’} # Two days ago
Get-<Noun> -Filter { StartTime -ge ‘-1:30’} # Hour and a half ago
```


Get-<Noun> -Filter { StartTime -ge '-0:0:30'} # 30 seconds ago

Array Properties

When filtering against list or array properties, simple parameters perform a case-insensitive wildcard match against each of the members. With filter expressions, you can use the -contains and -notcontains operators. Unlike PowerShell, these perform wildcard matching on strings.
Note that for array properties the naming convention is for the returned property to be plural, but the parameter used to search for any match is singular. The following are equivalent (assuming Users is an array property):

Get-<Noun> -User Fred*
Get-<Noun> -Filter { User -like "Fred*"}
Get-<Noun> -Filter { Users -contains "Fred*"}

You can also use the singular form with -Filter to search using other operators:

```
# Match if any user in the list is called "Frederick"
Get-<Noun> -Filter {
  User -eq "Frederick"  }

# Match if any user in the list has a name alphabetically below 'F'
Get-<Noun> -Filter {
  User -lt 'F'  }
```

Complex Expressions

When matching against multiple values, you can use a sequence of comparisons joined with -or operators, or you can use -in and -notin:

Get-<Noun> -Filter { Shape -eq 'Circle'-or Shape -eq 'Square'}
$shapes = 'Circle','Square'
Get-<Noun> -Filter { Shape -in $shapes }
$sides = 1..4
Get-<Noun> -Filter { Sides -notin $sides }

Braces can be used to group complex expressions, and override the default left-to-right evaluation of -and and -or. You can also use -not to invert the sense of any sub-expression:

Get-<Noun> -Filter { Size -gt 4 -or (Color -eq 'Blue'-and Shape -eq 'Circle') }
Get-<Noun> -Filter { Sides -lt 5 -and -not (Color -eq 'Blue'-and Shape -eq 'Circle') }

Paging

Citrix recommends that you avoid paging by using *Properties* or the *-Filter* mechanism.

However, if more objects are required, then the SDK supports deterministic paging though filtering. You must sort and filter objects to have a unique identifier for an object. This unique identifier helps you to page through objects and avoid reordering. Most of the Broker objects have a unique ID that can be used for paging. However, you must sort the other objects that do not have a unique ID either by time or some other unique field.

Example:

```
# Paging example
$allSessions = @()
$lastUid = 0
while ($true)
{

  $sessions = @(Get-BrokerSession -Filter {
  Uid -gt $lastUid  }
  -MaxRecordCount 1000 -Sortby 'Uid')
```

if ($sessions.Length -eq 0)
{

break;

}
$lastUid = $sessions[-1].Uid
$allSessions += $sessions
}

In this example, a list is created and the lowest possible value for the chosen unique field is initialized. The unique ID in this example is zero. In a loop, sessions are requested using a filter searching for objects
with a unique ID greater than zero. The number of objects is restricted to the first 1000 entries.
It is important to sort by the field that is used as a filter.

The filtering UID ($lastUid) is set to the unique ID of the final object retrieved.
The loop begins again using this unique ID value as the lower bound.
This loop continues until all sessions are retrieved and stored in an array ($allSessions).

Filter Syntax Definition

<Filter> ::= <ScriptBlock> | <ComponentList>

<ScriptBlock> ::= “{“<ComponentList> “}”

<ComponentList> ::= <Component> <AndOrOperator> <ComponentList> |

<Component>

<Component> ::= <NotOperator> <Factor> |

<Factor>

<Factor> ::= “(“<ComponentList> “)”|

<PropertyName> <ComparisonOperator> <Value> |
<PropertyName>

<AndOrOperator> ::= “-and”| “-or”

<NotOperator> ::= “-not”| “!”

<ComparisonOperator>

::= “-eq”| “-ne”| “-le”| “-ge”| “-lt”| “-gt”|

“-like”| “-notlike”| “-contains”| “-notcontains”|
“-in”| “-notin”

<PropertyName> ::= <simple name of property>

<Value> ::= <string literal> | <numeric literal> |

<scalar variable> | <array variable> |
“$null”| “$true”| “$false”

Numeric literals support decimal and hexadecimal literals, with optional multiplier suffixes (kb, mb, gb, tb, pb).

Dates and times can be specified as string literals. The current culture determines what formats are accepted. To avoid any ambiguity, use strings formatted to the ISO8601 standard. If not specified, the current time zone is used.

Relative date-time string literals are also supported, using a minus sign followed by a TimeSpan. For example, “-1:30”means 1 hour and 30 minutes ago.

about_XenDesktopModule

March 11, 2024

Topic

about_XenDesktopModule

Short Description

The XenDesktop PowerShell module provides high level administrative functions for the Citrix XenDesktop.

Command Prefix

All commands in this module have the noun prefixed with 'XD'.

Long Description

The XenDesktop PowerShell module enables both local and remote administration for Citrix XenDesktop. It provides facilities to configure and administer a XenDesktop site.

TODO

about_XenDesktopModule_SiteConfiguration

March 11, 2024

Topic

XenDesktop PowerShell Module - Site Configuration

Short Description

Describes how to do the initial configuration of a Site using the XenDesktop PowerShell Module.

Command Prefix

All commands in this module have 'XD'in their name.

Long Description

After installing XenDesktop for the first time, the Site must be configured. Site configuration involves:

- Creating the databases that will be used by the Site.
- Instructing the various XenDesktop services to use the databases and making those services aware of each other.
- Licensing the Site and specifying which edition of XenDesktop to use.

Additional Delivery Controllers may be added to the Site once the initial configuration is complete.

Database Creation

There is one main database, namely the Site Configuration Database, and two secondary databases, namely the Configuration Logging Database and the Monitoring Database. These databases may be hosted on the same SQL Server or on dedicated SQL Servers.

The method used to create the databases depends on the SQL Server permissions held by the person configuring the Site, whether or not the databases use default names, and whether or not the databases are hosted on the same SQL Server.

If the person configuring the Site does not have sufficient privilege to create databases and add database logons in the SQL Server, then database creation is a two-stage process:

- Use the Get-XDDatabaseSchema cmdlet to generate one or more SQL scripts that contain commands to create the databases and configure SQL logons.
- The scripts must be executed by a database administrator with suitable privileges on the target SQL Server(s).

If the person configuring the Site has sufficient privilege, database creation is a single-stage process in which the New-XDDatabase cmdlet is used to create the databases and configure SQL logons.

When non-default database names are to be used, or the databases are to be hosted on individual SQL Servers, you must run either Get-XDDatabaseSchema or New-XDDatabase cmdlets once for each database.

Database Mirroring

There are two variations of the New-XDDatabase cmdlet:

- One that is intended for Proof of concept and basic deployments. This variation is defined by the -AllDefaultDatabases switch parameter.
- The other is used to configure a database for each data store (Site, Logging, and Monitoring) individually. Each data store can be configured on different database server with different levels of access (user credentials). It is on these environments that mirroring is expected to be configured because mirroring can be targeted at a specific datastore. For instance; it may be considered that mirroring is not important for the monitoring data store.

When New-XDDatabase is to be used to configure each data store separately an empty database should be created on the database server with mirroring enabled before running New-XDDatabase. The empty database must have a collation which ends with "_CI_AS_KS". In general, it is best to use a collation which ends with "_100_CI_AS_KS ". Once the databases have been configured for mirroring call New-XDDatabase for each data store (See the -DataStore parameter) and supply the principle database server address, the database credentials and the database name. New-XDDatabase will just populate the empty database if it sees that it is there. The mirror server address will be detected automatically and login scripts uploaded to the mirror server to allow the XenDesktop services access to the mirror database upon failover.

If using the "-AllDefaultDatabases"parameter set of these cmdlets the database names will not be known to the user upfront. Mirroring will need to be configured on the databases after they have been created using %5BNew-XDDatabase%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/XenDesktop/New-XDDatabase.html). %5BNew-XDDatabase%5D(/en-us/citrix-virtual-apps-desktops-sdk/2311/XenDesktop/New-XDDatabase.html) will return information about the database names once it has completed. To configure mirroring on the databases; go to the SQL server and configure mirroring on the databases in the usual way. The mirror server will also need updating to allow XenDesktop access upon failover. To do this call:

"Get-XDDatabaseSchema -ScriptType AddDatabaseLogOn -DataStore …"

for each datastore. Append the returned scripts together and upload the scripts to the mirror server. If this step is not performed XenDesktop will

not be able to connect to the mirror server. The mirror server address can then be supplied to New-XDSite.

The databases can be created entirely via a manual process too. In which case New-XDDatabse will not be used. Instead call:

"Get-XDDatabaseSchema -ScriptType FullDatabase -DataStore …"

for each data store and upload the resultant script to the database on the principle database server. Configure mirroring on the database if required and call:

"Get-XDDatabaseSchema -ScriptType AddDatabaseLogOn -DataStore …"

and upload the resultant script to the mirror server. This will allow XenDesktop access to the server upon failover.

Once the databases have been created and populated call New-XDSite and supply the full information (Database Name, Database Server Address, and Database Mirror Server Address) for each data store.

Service Initialization

During service initialization the XenDesktop services are instructed to use the databases, Configuration Logging is started and a temporary license is automatically enabled using the New-XDSite cmdlet. If SQL Mirroring is enabled for the databases, the services are also made aware of the SQL Mirror Servers.

The site will run in a grace period until Set-XDLicensing is used to configure licensing.

There are three variations of the New-XDSite cmdlet to handle the following scenarios:

- The databases were created with default names on a single SQL Server.
- The databases were created with non-default names on a single SQL Server.
- The databases were created on individual SQL Servers.

Licensing

The product edition and licensing model of the Site must be set, and details of a Citrix License Server from which licenses can be obtained must be provided. Licensing must be set after running New-XDSite. Use the Set-XDLicensing cmdlet.

Adding Controllers

The method used to add a Controller to an existing Site depends on the SQL Server permissions held by the person adding the Controller.

If the person adding the Controller does not have sufficient privilege to alter content of the databases and add database logons, then Controller addition is a three-stage process:

- The Get-XDDatabaseSchema cmdlet is used to generate one or more SQL scripts that contain commands to alter database content and configure SQL logons.
- The scripts must be executed by a database administrator with suitable privileges on the target SQL Server(s).
- After the scripts have been executed, the Add-XDController cmdlet is used with the DoNotUpdateDatabaseServer switch to configure the XenDesktop services on the new Controller.

If the person adding the Controller has sufficient privilege, the Add-XDController cmdlet is used without the DoNotUpdateDatabaseServer switch to alter database content, configure SQL logons and configure the XenDesktop services on the new Controller.

Removing Controllers

The method used to remove a Controller from a Site depends on the SQL Server permissions held by the person removing the Controller.

If the person removing the Controller does not have sufficient privilege to alter content of the databases and remove database logons, then Controller removal is a three-stage process:

- Use the Remove-XDController cmdlet with the DoNotUpdateDatabaseServer switch to decommission the XenDesktop services on the Controller that is to be removed.
- Use the Get-XDDatabaseSchema cmdlet to generate one or more SQL scripts that contain commands to alter database content and remove SQL logons.
- Execute the scripts on the target SQL Server(s). This must be done by a database administrator with suitable privileges.

If the person removing the Controller has sufficient privilege, the Remove-XDController cmdlet is used without the DoNotUpdateDatabaseServer switch to decommission the XenDesktop services on the new Controller, alter database content and remove SQL logons.

Once a Controller has been removed from a Site, that Controller may be retired or added to another Site.

Changing Secondary Databases

This may be required when, for example, the data contained in the

Monitoring Database Has Grown To Such An Extent That A Dedicated

Sql Server Is Needed. The Method Required To Change The Name Of

One Of The Secondary Databases That Supports Configuration Logging

and Monitoring, or the SQL Server hosting one of those databases, depends on the SQL Server permissions held by the person changing the database.

If the person changing a secondary database does not have sufficient privilege to create databases in the SQL Server hosting the new secondary databases and add database logons, then modification is a three-stage process:

- Use the Get-XDDatabaseSchema cmdlet to generate an SQL script that contains commands to create the secondary database and configure SQL logons.
- Execute the script on the target SQL Server. This must be done by a database administrator with suitable privileges.
- After executing the script, use the Set-XDLogging and/or Set-XDMonitor cmdlets to configure the XenDesktop services in the Site to use the new secondary databases.

If the person changing a secondary database has sufficient privilege to create databases in the SQL Server hosting the new secondary databases and add database logons, then:

- Use the New-XDDatabase cmdlet to create the secondary databases and configure SQL logons.

- Use the Set-XDLogging and/or Set-XDMonitor cmdlets to configure the XenDesktop services in the Site to use the new secondary databases.

Examples

Single Sql Server With Default Database Names, Sql Admin Access And No Sql

Mirroring

You have full administrative rights on the SQL Server 'SqlServer1'with instance 'SQLEXPRESS'and want to create a Site called 'Site1'and associate it with the Controller at 'Controller1'. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer.

```
# Create the databases
 New-XDDatabase -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -AllDefaultDatabases -SiteName Site1
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -AllDefaultDatabases -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer
```

Create A Site Using A Single Sql Server With Default Database Names, Sql Admin

Access And Sql Mirroring

You have full administrative rights on the SQL Server 'SqlServer1'with instance 'Instance1'and want to create a Site called 'Site1'and associate it with the Controller at 'Controller1'. You want to configure the databases for SQL Server Mirroring using 'SqlMirror1'with 'Instance1'. Your Citrix License Server is listening on the non default port 27001 of server CitrixLicenseServer.

```
# Create the databases
 New-XDDatabase -AdminAddress Controller1 -DatabaseServer
SqlServer1\Instance1 -AllDefaultDatabases -SiteName Site1
```

```
#
# Configure your SQL Server Mirror in the normal manner to mirror the
    three
```



```
# default databases
#
```

```
# Create the Site and provide the mirror information
New-XDSite -AdminAddress Controller1 -DatabaseServer
SqlServer1\Instance1 -DatabaseMirrorServer SqlMirror1\Instance1
-AllDefaultDatabases -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -LicenseServerPort 27001
```

Create A Site Using A Single Sql Server With Default Database Names, Sql Admin Access Under Separate Username And No Sql Mirroring

You have a separate user account with full administrative rights on the SQL Server 'SqlServer1'with instance 'SQLEXPRESS'and want to create a Site called 'Site1'and associate it with the Controller at 'Controller1'. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer and you have a Platinum license.

```
# Construct a Credential object for your SQL Administrator
 $cred = Get-Credential
```

```
# Create the databases
 New-XDDatabase -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -DatabaseCredentials $cred -AllDefaultDatabases
-SiteName Site1
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -AllDefaultDatabases -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -ProductEdition PLT
```

Create A Site Using A Single Sql Server With Non-Default Database Names, Sql Admin Access And No Sql Mirroring

You have full administrative rights on the SQL Server 'SqlServer1'with instance 'SQLEXPRESS'and want to create a Site called 'Site1', associate it with the Controller at 'Controller1'and use database names of 'CitrixSiteDb',

‘CitrixLoggingDb’and ‘CitrixMonitorDb’. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer and you have an Enterprise license.

```
# Create the Site Database
New-XDDatabase -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixSiteDb
-SiteName Site1
```

```
# Create the Logging Database
New-XDDatabase -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixLoggingDb
-SiteName Site1
```

```
# Create the Monitor Database
New-XDDatabase -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixMonitorDb
-SiteName Site1
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -SiteDatabaseName CitrixSiteDb -
    LoggingDatabaseName
CitrixLoggingDb -MonitorDatabaseName CitrixMonitorDb -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -ProductEdition ENT
```

Create A Site Using One Sql Server Per Database With Non-Default Database

Names, Sql Admin Access And No Sql Mirroring

You have full administrative rights on the SQL Servers ‘SqlServerSite’, ‘SqlServerLogging’and ‘SqlServerMonitor’with instance ‘SQLEXPRESS’on each SQL Server. You want to create a Site called ‘Site1’, associate it with the Controller at ‘Controller1’and use database names of ‘CitrixSiteDb’, ‘CitrixLoggingDb’and ‘CitrixMonitorDb’. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer and you have a Platinum license.

```
# Create the Site Database
New-XDDatabase -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServerSite\SQLEXPRESS -DatabaseName CitrixSiteDb
-SiteName Site1
```

```
# Create the Logging Database
```



```
New-XDDatabase -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServerLogging\SQLEXPRESS -DatabaseName
   CitrixLoggingDb
-SiteName Site1
```

```
# Create the Monitor Database
New-XDDatabase -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServerMonitor\SQLEXPRESS -DatabaseName
   CitrixMonitorDb
-SiteName Site1
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -SiteDatabaseServer
SqlServerSite\SQLEXPRESS -SiteDatabaseName CitrixSiteDb
-LoggingDatabaseServer SqlServerLogging\SQLEXPRESS -LoggingDatabaseName
CitrixLoggingDb -MonitorDatabaseServer SqlServerMonitor\SQLEXPRESS
-MonitorDatabaseName CitrixMonitorDb -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -ProductEdition PLT
```

Create A Site Using One Sql Server Per Database With Non-Default Database

Names, Sql Admin Access And Sql Mirroring

You have full administrative rights on the SQL Servers ‘SqlServerSite’, ‘SqlServerLogging’and ‘SqlServerMonitor’with instance ‘Instance1’on each SQL Server. You want to create a Site called ‘Site1’, associate it with the Controller at ‘Controller1’and use database names of ‘CitrixSiteDb’, ‘CitrixLoggingDb’and ‘CitrixMonitorDb’. You also want to configure the databases for SQL Server Mirroring using ‘SqlMirrorSite’, ‘SqlMirrorLogging’and ‘SqlMirrorMonitor’with instance ‘Instance1’. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer and you have a Platinum license.

```
#
# Create the empty databases for each data store and configure
   mirroring
# on those databases.
#
```

```
# Populate the Site Database
New-XDDatabase -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServerSite\Instance1 -DatabaseName CitrixSiteDb
-SiteName Site1
```



```
# Populate the Logging Database
New-XDDatabase -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServerLogging\Instance1 -DatabaseName
    CitrixLoggingDb
-SiteName Site1
```

```
# Populate the Monitor Database
New-XDDatabase -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServerMonitor\Instance1 -DatabaseName
    CitrixMonitorDb
-SiteName Site1
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -SiteDatabaseServer
SqlServerSite\Instance1 -SiteDatabaseMirrorServer SqlMirrorSite\
    Instance1
-SiteDatabaseName CitrixSiteDb -LoggingDatabaseServer
SqlServerLogging\Instance1 -LoggingDatabaseMirrorServer
SqlMirrorLogging\Instance1 -LoggingDatabaseName CitrixLoggingDb
-MonitorDatabaseServer SqlServerMonitor\Instance1
-MonitorDatabaseMirrorServer SqlMirrorMonitor\Instance1
-MonitorDatabaseName CitrixMonitorDb -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -ProductEdition PLT
```

Create A Site Using A Single Sql Server With Non-Default Database Names, No Sql Admin Access And No Sql Mirroring

You have no administrative rights on the SQL Server ‘SqlServer1’with instance ‘SQLEXPRESS’and want to create a Site called ‘Site1’, associate it with the Controller at ‘Controller1’and use database names of ‘CitrixSiteDb’, ‘CitrixLoggingDb’and ‘CitrixMonitorDb’. Your Citrix License Server is listening on the default port (27000) of server CitrixLicenseServer and you have a Platinum license.

```
# Get the SQL script to create the Site Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixSiteDb -
    SiteName
Site1 -ScriptType FullDatabase > C:\siteScript.sql
```

```
# Get the SQL script to create the Logging Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixLoggingDb
```



```
-SiteName Site1 -ScriptType FullDatabase > C:\LoggingScript.sql
```

```
# Get the SQL script to create the Monitor Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixMonitorDb
-SiteName Site1 -ScriptType FullDatabase > C:\monitorScript.sql
```

```
#
# Get an individual with suitable privileges to execute the SQL scripts
# 'siteScript.sql', 'loggingScript.sql' and 'monitorScript.sql'on
# 'SqlServer1'
#
```

```
# Create the Site
New-XDSite -AdminAddress Controller1 -DatabaseServer
SqlServer1\SQLEXPRESS -SiteDatabaseName CitrixSiteDb -
    LoggingDatabaseName
CitrixLoggingDb -MonitorDatabaseName CitrixMonitorDb -SiteName Site1
```

```
# Set licensing
Set-XDLicensing -AdminAddress Controller1 -LicenseServerAddress
CitrixLicenseServer -ProductEdition PLT
```

Determine Applicable Licensing Models And Product Editions

You want to determine the license models and product editions that are applicable to your XenDesktop installation.

```
# Determine your product code
$site = Get-XDSite
$productCode = $site.LicenseInformation.ProductCode
```

```
# Load the Citrix.Configuration.Admin.V2 snapin, as this exposes
    cmdlets to
# retrieve the licensing models and product editions
Add-PSSnapin Citrix.Configuration.Admin.V2
```

```
# Get the applicable licensing models
Get-ConfigLicensingModel -ProductCode $productCode
```

```
# Get the applicable product editions
Get-ConfigProductEdition -ProductCode $productCode
```

Change The Configuration Logging Database

You have full administrative rights on the SQL Server ‘SqlServer2’with instance ‘INSTANCE1’and want to change Configuration Logging for site ‘Site1’to

use a new database called ‘CitrixLoggingDb_New’on that server.

```
# Create the Configuration Logging Database on the new SQL Server
New-XDDatabase -Datastore Logging -SiteName Site1 -DatabaseServer
SqlServer2\INSTANCE1 -DatabaseName CitrixLoggingDb_New
```

```
# Change Configuration Logging to use the new database
Set-XDLogging -DatabaseServer SqlServer2\INSTANCE1 -DatabaseName
CitrixLoggingDb_New
```

Change The Monitoring Database

You have full administrative rights on the SQL Server ‘SqlServer2’with instance ‘INSTANCE1’and want to change Monitoring for site ‘Site1’to use a new database called ‘CitrixMonitorDb_New’on that server.

```
# Create the Monitoring Database on the new SQL Server
New-XDDatabase -Datastore Monitor -SiteName Site1 -DatabaseServer
SqlServer2\INSTANCE1 -DatabaseName CitrixMonitorDb_New
```

```
# Change Monitoring to use the new database
Set-XDMonitor -DatabaseServer SqlServer2\INSTANCE -DatabaseName
CitrixMonitorDb_New
```

Add A Controller To A Site For Which You Have Full Db Admin

You have full administrative rights on the SQL Server that underlies a Site called ‘Site1’. An existing Controller in the Site is ‘Controller1’and you want to add Controller ‘AdditionalController’to the Site.

```
Add-XDController -AdminAddress AdditionalController
-SiteControllerAddress Controller1
```

Add A Controller To A Site For Which You Have Full Db Admin Under Separate Username

You have a separate user account with full administrative rights on the SQL Server that underlies a Site called ‘Site1’. An existing Controller in the Site is ‘Controller1’and you want to add Controller ‘AdditionalController’to the Site.

```
# Construct a Credential object for your SQL Administrator
 $cred = Get-Credential
```



```
# Add the Controller
Add-XDController -AdminAddress AdditionalController
-SiteControllerAddress Controller1 -DatabaseCredentials $cred
```

Add A Controller To A Site For Which You Have Full Db Admin Under Separate

Usernames

You have separate user accounts with full administrative rights on the three SQL Servers that underlie a Site called 'Site1'. An existing Controller in the Site is 'Controller1'and you want to add Controller 'AdditionalController' to the Site.

```
# Construct a Credential object for your Site Database SQL
    Administrator
 $credSite = Get-Credential
```

```
# Construct a Credential object for your Logging Database SQL
    Administrator
 $credLogging = Get-Credential
```

```
# Construct a Credential object for your Monitor Database SQL
    Administrator
 $credMonitor = Get-Credential
```

```
# Add the Controller
Add-XDController -AdminAddress AdditionalController
-SiteControllerAddress Controller1 -SiteDatabaseCredentials $credSite
-LoggingDatabaseCredentials $credLogging -MonitorDatabaseCredentials
$credMonitor
```

Add A Controller To A Site For Which You Have No Db Admin Access

You have no administrative rights on the SQL Server 'SqlServer1'with instance 'SQLEXPRESS'that underlies a Site called 'Site1'. The Site's databases are called 'CitrixSiteDb', 'CitrixLoggingDb'and 'CitrixMonitorDb'. An existing Controller in the Site is 'Controller1'and you want to add Controller 'AdditionalController'to the Site.

```
# Get the SQL script that may be used to add permissions for
# 'AdditionalController' to the Site Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixSiteDb -
    SiteName
Site1 -ScriptType AddController > C:\siteScript.sql
```



```
# Get the SQL scripts that may be used to add permissions for '
# AdditionalController' to the Logging Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixLoggingDb
-SiteName Site1-ScriptType AddController > C:\loggingScript.sql
```

```
# Get the SQL script that may be used to add permissions for
# 'AdditionalController' to the Monitor Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixMonitorDb
-SiteName Site1-ScriptType AddController > C:\monitorScript.sql
```

```
#
# Get an individual with suitable privileges to execute the SQL scripts
# 'siteScript.sql', 'loggingScript.sql' and 'monitorScript.sql'on
# 'SqlServer1'
#
```

```
# Add the Controller
Add-XDController -AdminAddress AdditionalController
-SiteControllerAddress Controller1 -DoNotUpdateDatabaseServer
```

Remove A Controller From A Site For Which You Have Full Db Admin Access

You have full administrative rights on the SQL Server that underlies a Site called ‘Site1’. The Site currently has two Controllers, namely ‘Controller1’and ‘RedundantController’. You want to remove ‘RedundantController’from the Site.

```
Remove-XDController -AdminAddress Controller1 -ControllerName
RedundantController
```

Remove A Controller From A Site For Which You Have Full Db Admin Under

Separate Username

You have a separate user account with full administrative rights on the SQL Server that underlies a Site called ‘Site1’. The Site currently has two Controllers, namely ‘Controller1’and ‘RedundantController’. You want to remove ‘RedundantController’from the Site.

```
# Construct a Credential object for your SQL Administrator
 $cred = Get-Credential
```

```
# Remove the Controller
```



```
Remove-XDController -AdminAddress Controller1 -ControllerName
RedundantController -DatabaseCredentials $cred
```

Remove A Controller From A Site For Which You Have Full Db Admin Under

Separate Usernames

You have separate user accounts with full administrative rights on the three SQL Servers that underlie a Site called 'Site1'. The Site currently has two Controllers, namely 'Controller1'and 'RedundantController'. You want to remove 'RedundantController'from the Site.

```
# Construct a Credential object for your Site Database SQL
    Administrator
 $credSite = Get-Credential
```

```
# Construct a Credential object for your Logging Database SQL
    Administrator
 $credLogging = Get-Credential
```

```
# Construct a Credential object for your Monitor Database SQL
    Administrator
 $credMonitor = Get-Credential
```

```
# Remove the Controller
Remove-XDController -AdminAddress Controller1 -ControllerName
RedundantController -SiteDatabaseCredentials $credSite
-LoggingDatabaseCredentials $credLogging -MonitorDatabaseCredentials
$credMonitor
```

Remove A Controller From A Site For Which You Have No Db Admin Access

You have no administrative rights on the SQL Server 'SqlServer1'with instance 'SQLEXPRESS'that underlies a Site called 'Site1'. The Site's databases are called 'CitrixSiteDb', 'CitrixLoggingDb'and 'CitrixMonitorDb'. The Site currently has two Controllers, namely 'Controller1'and 'RedundantController'. You want to remove 'RedundantController'from the Site. 'RedundantController'is in the 'ACME'domain.

```
# Determine the SID of the Controller that is to be removed.
$site = Get-XDSite
$redundantController = $site.Controllers  | Where-Object
{
 $_.MachineName -eq ' ACME\RedundantController ' }

```



```
$redundantControllerSid = $redundantController.Sid
```

```
# Remove the Controller
Remove-XDController -AdminAddress Controller1 -ControllerName
RedundantController -DoNotUpdateDatabaseServer
```

```
# Get the SQL scripts that may be used to remove permissions for
# 'RedundantController' from the Site Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Site
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixSiteDb -
    SiteName
Site1 -ScriptType RemoveController -ControllerToRemove
$redundantControllerSid > C:\siteScript.sql
```

```
# Get the SQL script that may be used to remove permissions for '
# RedundantController' from the Logging Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Logging
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixLoggingDb
-SiteName Site1-ScriptType RemoveController -ControllerToRemove
$redundantControllerSid > C:\loggingScript.sql
```

```
# Get the SQL script that may be used to remove permissions for
# 'RedundantController' from the Monitor Database
Get-XDDatabaseSchema -AdminAddress Controller1 -DataStore Monitor
-DatabaseServer SqlServer1\SQLEXPRESS -DatabaseName CitrixMonitorDb
-SiteName Site1-ScriptType RemoveController -ControllerToRemove
$redundantControllerSid > C:\monitorScript.sql
```

```
#
# Although not necessary, Citrix recommends you get an individual
# with suitable privileges to execute the SQL scripts 'siteScript.sql',
# 'loggingScript.sql' and 'monitorScript.sql'on 'SqlServer1'
```


See Also

Get-Xddatabaseschema

New-Xddatabase

New-Xdsite

Set-Xdlicensing

Add-Xdcontroller

Remove-Xdcontroller

Set-Xdlogging

Set-Xdmonitor

Add-XDController

March 11, 2024

Adds a Delivery Controller to an existing Site.

Syntax

```
Add-XDController
   [-DoNotUpdateDatabaseServer]
   -SiteControllerAddress <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

```
Add-XDController
   -DatabaseCredentials <PSCredential>
   -SiteControllerAddress <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

```
Add-XDController
   -LoggingDatabaseCredentials <PSCredential>
   -MonitorDatabaseCredentials <PSCredential>
   -SiteControllerAddress <String>
   -SiteDatabaseCredentials <PSCredential>
   [-AdminAddress <String>]
```



```
7     [<CommonParameters>]
```

Description

Adds a Controller to an existing Site.

If no credentials are provided and DoNotUpdateDatabaseServer has not been specified, then all database update operations are attempted using the credentials of the user running the cmdlet.

If DatabaseCredentials is provided, then all database operations are attempted using those credentials.

If one or more of SiteDatabaseCredentials, LoggingDatabaseCredentials or MonitorDatabaseCredentials are provided, then all three must be provided. All operations are then attempted using the credentials appropriate to each Database.

Examples

EXAMPLE 1

Adds the Controller 'MyNewController'to the Site to which 'MyExistingController'is already joined. The permissions for 'MyNewController'will be added automatically to the Database using the implicit credentials of the user running the cmdlet.

```
1 Add-XDController -AdminAddress MyNewController -SiteControllerAddress
     MyExistingController
```

EXAMPLE 2

Adds the Controller 'MyNewController'to the Site to which 'MyExistingController'is already joined.

The permissions for 'MyNewController'will not be added to the Database. Instead the Get-XDDatabaseSchema cmdlet is used to generate a SQL script that must be given to the database administrator of 'MySQLServer'. The SQL script must be used to add the necessary permissions for 'MyNewController'before Add-XDController is run.

```
1 Get-XDDatabaseSchema -AdminAddress MyNewController -SiteName MySite -
     DatabaseServer MySQLServer -ScriptType AddController > C:\
     addController.sql
2 # Get a database administrator to run addController.sql
3 Add-XDController -AdminAddress MyNewController -SiteControllerAddress
     MyExistingController -DoNotUpdateDatabaseServer
```


EXAMPLE 3

Adds the Controller 'MyNewController'to the Site to which 'MyExistingController'is already joined.

The credentials contained within '$credential'will be used to add the necessary permissions for 'MyNewController'to the Database.

```
$credential = Get-Credential
Add-XDController -AdminAddress MyNewController -SiteControllerAddress
    MyExistingController -DatabaseCredentials $credential
```

EXAMPLE 4

Adds the Controller 'MyNewController'to the Site to which 'MyExistingController'is already joined.

The credentials contained within '$credentialSite'will be used to add the necessary permissions for 'MyNewController'to the Site Configuration Database. The credentials contained within '$credentialLogging'will be used to add the necessary permissions for 'MyNewController'to the Configuration Logging Database. The credentials contained within '$credentialMonitor'will be used to add the necessary permissions for 'MyNewController'to the Monitoring Database.

```
$credentialSite = Get-Credential
$credentialLogging = Get-Credential
$credentialMonitor = Get-Credential
Add-XDController -AdminAddress MyNewController -SiteControllerAddress
    MyExistingController -SiteDatabaseCredentials $credentialSite -
    LoggingDatabaseCredentials $credentialLogging -
    MonitorDatabaseCredentials $credentialMonitor
```

Parameters

-DatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Servers associated with the Site Configuration, Configuration Logging and Monitoring Databases.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Configuration Logging Database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Monitoring Database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Site Configuration Database.

Type:	PSCredential
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteControllerAddress

The address of a Controller in the Site to which this Controller is to be joined.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DoNotUpdateDatabaseServer

Results in the permissions associated with the Controller not being automatically added to the Database.

This switch may be useful when the person running the cmdlet does not have sufficient database privilege. The database permissions for the Controller that is to be added must have been applied manually prior to running this cmdlet, otherwise the Controller is not added and an error is returned.

Get-XDDatabaseSchema may be used to generate the SQL script which in turn may be used to add the necessary permissions to the Database.

This parameter results in all supplied credentials being ignored.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False

Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

In general, the AdminAddress specifies the Controller to which the PowerShell module will connect. However, in this cmdlet, not only is this the address of the Controller to connect to, it is also the Controller that is to be added to this Site.

If the user running this cmdlet does not have privilege to add users to the Database, then use either the DoNotudpateDatabaseServer switch or supply alternative credentials using one or more of the Credential arguments.

The command can fail for the following reasons:

- The Controller is already a member of a Site.
- The DoNotUpdateDatabaseServer parameter was specified, but the permissions for the Controller are not already in the Database. Either add the permissions using Get-XDDatabaseSchema or allow this cmdlet to add the permissions automatically.
- The person running the cmdlet, or the supplied credentials, do not have sufficient privilege to update the Database. Either run this cmdlet as a user that has administrator privilege to the Database, or supply an account with suitable privileges using the PSCredential parameter(s).

Related Links

- Get-XDDatabaseSchema
- Get-XDSite
- Remove-XDController

Get-XDDatabaseSchema

March 11, 2024

Gets the SQL scripts used to create and manage the Database.

Syntax

```
Get-XDDatabaseSchema
    [-DatabaseName <String>]
    -DatabaseServer <String>
    -DataStore <DataStore>
    [-ScriptType <DatabaseScriptType>]
    -SiteName <String>
    [-AdminAddress <String>]
    [<CommonParameters>]
```


Description

Generates SQL scripts that can be given to a database administrator for execution. Scripts are written to standard output.

A script of type FullDatabase is used to manually create the Database with write access permissions for the Delivery Controller identified by AdminAddress. This script must be generated and executed for each datastore type supported by the Site.

Once the database administrator has executed all scripts of type FullDatabase, run New-XDSite to configure the databases for first use.

Examples

EXAMPLE 1

Gets the script for creating the Site Configuration Database and writes the content to the file "c:\script.sql". This script may be executed on a database server "MyDBServer"for an existing and empty database with name "MySiteDB".

```
Get-XDDatabaseSchema -SiteName MySiteServiceGroup -DataStore Site -
    DatabaseName MySiteDB -DatabaseServer MyDBServer -ScriptType
    FullDatabase > c:\script.sql
```

EXAMPLE 2

Gets the script used to write the access permissions for 'MyNewController'to the Configuration Logging Database. The contents of the script are written to the file "c:\script.sql". This script may be executed on a database server "MyDBServer"for an existing database with name "MyLoggingDB". The Controller can then only be added to the Site by running Add-XDController.

```
Get-XDDatabaseSchema -AdminAddress MyNewController -SiteName
    MySiteServiceGroup -DataStore Logging -DatabaseName MyLoggingDB -
    DatabaseServer MyDBServer -ScriptType AddController > c:\script.sql
```

EXAMPLE 3

Gets the script used to remove the access permissions for a Controller with SID 'S-1-5-21-1234567890-123456789-123456789-1234'from the Monitoring Database. The contents of the script are written to the file "c:\script.sql". This script can be executed only after the Controller has been removed from the Site with Remove-XDController. It may be executed on a database server "MyDBServer"for an existing database with name "MyLoggingDB".


```
Get-XDDatabaseSchema -SiteName MySiteServiceGroup -DataStore Monitor -
    DatabaseName MyMonitorDB -DatabaseServer MyDBServer -ScriptType
    RemoveController -ControllerToRemoveSid S
    -1-5-21-1234567890-123456789-123456789-1234 > c:\script.sql
```

EXAMPLE 4

Gets the script used to add a logon for 'MyExistingController'to the Site Configuration Database for the database mirror server "MyMirrorDBServer". The contents of the script are written to the file "c:\script.sql".

```
Get-XDDatabaseSchema -AdminAddress MyExistingController -SiteName
    MySiteServiceGroup -DataStore Site -DatabaseName MySiteDB -
    DatabaseServer MyMirrorDBServer -ScriptType AddDatabaseLogOn > c:\
    script.sql
```

Parameters

-DatabaseServer

The address of the SQL server on which the generated script will be run.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-DataStore

The datastore type of the script to be generated. There is no default value. Valid values are:

- Site

This corresponds to the Site Configuration Database.

- Logging

This corresponds to the Configuration Logging Database.

- Monitor

This corresponds to the Monitor Database.

Type:	DataStore
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteName

The name of the Site for which the SQL script is to be generated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	True

-DatabaseName

The name for the database of the selected datastore. There is a default name for each database. The default names are listed with their corresponding dataStore types as follows:

- Site

The Site Configuration Database. The default name of this database is Citrix<SiteName>.

- Logging

The Configuration Logging Database. The default name of this database is CitrixConfigLogging<SiteName>.

- Monitor

The Monitoring Database. The default name of this database is CitrixMonitor<SiteName>.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	True

-ScriptType

The type of SQL script to be generated. By default, a script of type FullDatabase is generated. Valid types are:

- AddController

This script is used to add a Controller specified by AdminAddress. It updates the specified database with all the access permissions needed by the Controller.

- AddDatabaseLogOn

This script is used to add a logon for a Controller to the specified database server. It is used specifically for configuring SQL Server mirroring, in which the mirror database server must have appropriate logons created for all Controllers in the site.

- FullDatabase

This script is used to create a database corresponding to the specified DataStore parameter. It also updates the database with all the access permissions needed for the Controller.

- RemoveController

This script is used to remove a Controller specified by the ControllerToRemoveSid parameter. It removes the database access permissions previously required by the Controller.

Type:	DatabaseScriptType
Position:	Named
Default value:	FullDatabase
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

String

A string containing the content of the generated SQL script.

Notes

When the ScriptType is set to RemoveController, you must specify the dynamic parameter "ControllerToRemoveSid". This specifies the SID of the Controller that is to be removed from the Site. The resultant database script removes the controller's permissions to change the Database data.

If a script of type FullDatabase is generated for a configured Site, the resulting script adds database access permissions for all known Controllers to the database to be created.

If the cmdlet fails, the following error codes can result:

- The datastore was not recognized.
- The database server name could not be resolved.

Related Links

- about_XenDesktopModule
- about_XenDesktopModule_SiteConfiguration
- New-XDDatabase
- Add-XDController
- Remove-XDController

Get-XDLicensing

March 11, 2024

Get the licensing attributes for a site.

Syntax

```
Get-XDLicensing
   [-AdminAddress <String>]
   [<CommonParameters>]
```

Description

Get the licensing attributes for the site which has a Controller identified by AdminAddress.

Examples

EXAMPLE 1

Get the licensing attributes of the Site which has a Controller identified by MyController.

```
Get-XDLicensing -AdminAddress MyController
```

Parameters

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.LicenseInformation

A Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.LicenseInformation object containing the relevant attributes.

Related Links

- Get-ConfigLicensingModel
- Get-ConfigProductEdition
- Get-XDSite
- Set-XDLicensing

Get-XDLogging

March 11, 2024

Gets the Configuration Logging Database attributes of a Site.

Syntax

```
Get-XDLogging
    [-AdminAddress <String>]
    [<CommonParameters>]
```

Description

Gets the Configuration Logging Database attributes of the Site which has a Controller identified by AdminAddress.

Examples

Parameters

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Logging

This includes whether Logging is enabled or not, whether configuration operations may proceed or not, even if the Configuration Logging Database is not available, the language used for Configuration Logging entries, the name of the Configuration Logging Database, the database server for the Configuration Logging Database, the mirror database server for the Configuration Logging Database and whether Integrated Security is enabled or not for the Configuration Logging Database.

Related Links

- Get-XDSite
- Set-XDLogging

Get-XDMonitor

March 11, 2024

Gets the Monitoring Database attributes of a Site.

Syntax

```
Get-XDMonitor
    [-AdminAddress <String>]
    [<CommonParameters>]
```

Description

Gets the Monitoring Database attributes of the Site which has a Controller identified by AdminAddress.

Examples

Parameters

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Monitor

This includes the name of the Monitoring Database, the database server for the Monitoring Database, the mirror database server for the Monitoring Database and whether Integrated Security is enabled or not for the Monitoring Database.

Related Links

- Get-XDSite
- Set-XDMonitor

Get-XDSite

March 11, 2024

Gets the attributes of a Site.

Syntax

```
Get-XDSite
    [-AdminAddress <String>]
    [<CommonParameters>]
```


Description

Gets the attributes of the Site which has a Controller identified by AdminAddress.

Examples

EXAMPLE 1

Gets the attributes of the Site which has a Controller identified by MySiteController.

```
Get-XDSite -AdminAddress MySiteController
```

Parameters

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Site

This includes the name of the Site, the Controllers in Site, the Databases supporting the Site, the licensing configuration of the Site and the default Icon for the Site.

Related Links

- about_XenDesktopModule_SiteConfiguration
- New-XDSite

New-XDDatabase

March 11, 2024

Creates the specified Database or all Databases.

Syntax

```
New-XDDatabase
   [-AllDefaultDatabases]
   [-DatabaseCredentials <PSCredential>]
   [-DatabaseNamePrefix <String>]
   -DatabaseServer <String>
   -SiteName <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

```
New-XDDatabase
   [-DatabaseCredentials <PSCredential>]
   [-DatabaseName <String>]
   -DatabaseServer <String>
   -DataStore <DataStore>
   -SiteName <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

Description

If AllDefaultDatabases is specified, this cmdlet creates all databases using their default names on the specified SQL server. When used in this manner, the cmdlet need only be called once.

If DataStore is specified, this cmdlet creates the specified database on the specified SQL Server. When used in this manner, the cmdlet should be called once per datastore.

This cmdlet creates, but does not configure, the Database. Run New-XDSite to configure the Site to use the Database.

For database mirroring please refer to "about_XenDesktopModule_SiteConfiguration".

Examples

EXAMPLE 1

Creates a Site Configuration Database called 'MySiteDatabase'for the Site named 'MySite'on 'Instance_1'of the SQL Server 'MySqlServer'.

```
New-XDDatabase -AdminAddress MySiteController -SiteName MySite -
    Datastore Site -DatabaseServer MySqlServer\Instance_1 -DatabaseName
    MySiteDatabase
```

EXAMPLE 2

Creates a Configuration Logging Database called 'MyLoggingDatabase'for the Site named 'MySite'on 'Instance_1'of the SQL Server 'MySqlServer'.

```
New-XDDatabase -AdminAddress MySiteController -SiteName MySite -
    Datastore Logging -DatabaseServer MySqlServer\Instance_1 -
    DatabaseName MyLoggingDatabase
```

EXAMPLE 3

Creates a Monitoring Database called 'MyMonitorDatabase'for the Site named 'MySite'on 'Instance_1'of the SQL Server 'MySqlServer'.

```
New-XDDatabase -AdminAddress MySiteController -SiteName MySite -
    Datastore Monitor -DatabaseServer MySqlServer\Instance_1 -
    DatabaseName MyMonitorDatabase
```

EXAMPLE 4

Creates Site Configuration, Configuration Logging and Monitoring Databases called 'MyPrefix-CitrixMySite', 'MyPrefix-CitrixConfigLoggingMySite'and 'MyPrefix-CitrixMonitorMySite'for the Site named 'MySite'on 'Instance_1'of the SQL Server 'MySqlServer'.


```
New-XDDatabase -AdminAddress MySiteController -SiteName MySite -
    AllDefaultDatabases -DatabaseServer MySqlServer\Instance_1 -
    DatabaseNamePrefix  MyPrefix-
```

Parameters

-AllDefaultDatabases

All databases are created with their default names. The default names are listed with their corresponding dataStore types as follows:

- Site

The Site Configuration Database. The default name of this database is Citrix<SiteName>.

- Logging

The Configuration Logging Database. The default name of this database is CitrixConfigLogging<SiteName>.

- Monitor

The Monitoring Database. The default name of this database is CitrixMonitor<SiteName>.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseServer

The address of the SQL Server in which the database is to be created. This must include the SQL Instance name.

Type:	String
Position:	Named
Default value:	None

Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteName

The name of the Site for which the specified database is to be created.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DataStore

The datastore type that is to be created. Valid values are Site, Logging and Monitor.

Type:	DataStore
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server specified by the DatabaseServer parameter. If this parameter is not provided, then the implicit credentials of the user running the cmdlet are used to connect to the SQL Server.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseNamePrefix

The prefix to be added to the default database names when AllDefaultDatabases is specified.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

The name that should be used for the database of the specified datastore type.

Type:	String
Position:	Named
Default value:	Site = Citrix, Logging = CitrixConfigLogging and Monitor = CitrixMonitor
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Database

One Database object is returned for each datastore type that is created. Each Database object includes the database name, datastore type and SQL Server address.

Notes

If the user running this cmdlet does not have privilege to create databases on the specified SQL Server, then alternative credentials may be provided using the DatabaseCredentials parameter.

The command can fail for the following reasons:

- A connection could not be established with the specified SQL Server.
- The person running the cmdlet, or the supplied credentials, do not have sufficient privilege to create databases on the specified SQL Server. In this situation, use Get-XDDatabaseSchema, instead of New-XDDatabase, to generate SQL database creation scripts that may be given to a database administrator for execution on the SQL Server.

Related Links

- about_XenDesktopModule
- about_XenDesktopModule_SiteConfiguration
- Get-XDDatabaseSchema
- Get-XDSite
- New-XDSite

New-XDSite

March 11, 2024

Configures the databases for use with a new Site and initializes all of the associated Site services.

Syntax

```
New-XDSite
   [-AllDefaultDatabases]
   [-DatabaseMirrorServer <String>]
   [-DatabaseNamePrefix <String>]
   -DatabaseServer <String>
   -SiteName <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

```
New-XDSite
   [-DatabaseMirrorServer <String>]
   -DatabaseServer <String>
   -LoggingDatabaseName <String>
   -MonitorDatabaseName <String>
   -SiteDatabaseName <String>
   -SiteName <String>
   [-AdminAddress <String>]
   [<CommonParameters>]
```

```
New-XDSite
```



```
    [-LoggingDatabaseMirrorServer <String>]
    -LoggingDatabaseName <String>
    -LoggingDatabaseServer <String>
    [-MonitorDatabaseMirrorServer <String>]
    -MonitorDatabaseName <String>
    -MonitorDatabaseServer <String>
    [-SiteDatabaseMirrorServer <String>]
    -SiteDatabaseName <String>
    -SiteDatabaseServer <String>
    -SiteName <String>
    [-AdminAddress <String>]
    [<CommonParameters>]
```

Description

Configures the Site for first use. This must be run after the databases have been created. The databases may have been created manually using the scripts generated from Get-XDDatabaseSchema or automatically using New-XDDatabase.

The site will run in a grace period until Set-XDLicensing is used to configure licensing.

For database mirroring please refer to "about_XenDesktopModule_SiteConfiguration".

Examples

EXAMPLE 1

Configures a site called 'MySite'for the Controller 'MyController'. All previously created databases have default names and are hosted on the SQL Server 'MySqlServer\Instance_1'.

```
New-XDSite -AdminAddress MyController -AllDefaultDatabases -
    DatabaseServer MySqlServer\Instance_1 -SiteName MySite
```

EXAMPLE 2

Configures a site called 'MySite'for the Controller 'MyController'. All previously created databases have default names, are hosted on the SQL Server 'MySqlServer'and mirrored on the SQL Server 'MySqlMirror'.

```
New-XDSite -AdminAddress MyController -AllDefaultDatabases -
    DatabaseServer MySqlServer -DatabaseMirrorServer MySqlMirror -
    SiteName MySite
```


EXAMPLE 3

Configures a site called 'MySite'for the Controller 'MyController'. All previously created databases have default names with a prefix of 'MyPrefix-'and are hosted on the SQL Server 'MySqlServer'.

```
New-XDSite -AdminAddress MyController -AllDefaultDatabases -
    DatabaseServer MySqlServer -SiteName MySite -DatabaseNamePrefix
    MyPrefix-
```

EXAMPLE 4

Configures a site called 'MySite'for the Controller 'MyController'. The previously created databases have non-default names of 'LoggingDb'for the Configuration Logging Database, 'MonitorDb'for Monitoring Database and 'SiteDb'for the Site Configuration Database. All databases are hosted on the SQL Server 'MySqlServer'.

```
New-XDSite -AdminAddress MyController -DatabaseServer MySqlServer -
    LoggingDatabaseName LoggingDb -MonitorDatabaseName MonitorDb -
    SiteDatabaseName SiteDb -SiteName MySite
```

EXAMPLE 5

Configures a site called 'MySite'for the Controller 'MyController'.

The previously created databases have non-default names and are hosted on individual SQL Servers. 'LoggingDb'and 'LoggingSqlServer 'for the Configuration Logging Database. 'MonitorDb'and 'MonitorSqlServer'for Monitoring Database. 'SiteDb'and 'SiteSqlServer'for the Site Configuration Database.

```
New-XDSite -AdminAddress MyController -LoggingDatabaseName LoggingDb -
    LoggingDatabaseServer LoggingSqlServer -MonitorDatabaseName
    MonitorDb -MonitorDatabaseServer MonitorSqlServer -SiteDatabaseName
    SiteDb -SiteDatabaseServer SiteSqlServer -SiteName MySite
```

Parameters

-AllDefaultDatabases

Indicates that all the databases that are to be used in this Site were created with their default names.

Type:	SwitchParameter
Position:	Named

Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseServer

If all databases were created within a single SQL Server, this is the address of that server. This must include the SQL Instance name.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteName

The name of the Site that is to be created.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDatabaseName

If databases were not created using their default names, this is the name of the Configuration Logging Database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorDatabaseName

If databases were not created using their default names, this is the name of the Monitoring Database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteDatabaseName

If databases were not created using their default names, this is the name of the Site Configuration Database.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDatabaseServer

If each database was created on a separate SQL Server, this is the name of the SQL Server hosting the Configuration Logging Database. This must include any applicable SQL Instance name.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorDatabaseServer

If each database was created on a separate SQL Server, this is the name of the SQL Server hosting the Monitoring Database. This must include any applicable SQL Instance name.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteDatabaseServer

If each database was created on a separate SQL Server, this is the name of the SQL Server hosting the Site Configuration Database. This must include any applicable SQL Instance name.

Type:	String
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseMirrorServer

If all databases were created within a single SQL Server and SQL Mirroring has been enabled for those databases, this is the address of the SQL Server that is acting as the Mirror. This must include the SQL Instance name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseNamePrefix

If a name prefix was applied to the default database names during the original creation of the databases, this specifies that name prefix.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDatabaseMirrorServer

If each database was created on a separate SQL Server and SQL Mirroring has been enabled for the the Configuration Logging Database, this is the address of the SQL Server that is acting as the mirror. This must include any applicable SQL Instance name.

Type:	String
Position:	Named
Default value:	If this parameter is not provided, the mirror server is determined by querying the SQL Server hosting the Configuration Logging Database.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorDatabaseMirrorServer

If each database was created on a separate SQL Server and SQL Mirroring has been enabled for the the Monitoring Database, this is the address of the SQL Server that is acting as the mirror. This must include any applicable SQL Instance name.

Type:	String
Position:	Named
Default value:	If this parameter is not provided, the mirror server is determined by querying the SQL Server that is hosting the Monitoring Database.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-SiteDatabaseMirrorServer

If each database was created on a separate SQL Server and SQL Mirroring has been enabled for the the Site Configuration Database, this is the address of the SQL Server that is acting as the mirror. This must include any applicable SQL Instance name.

Type:	String
Position:	Named

Default value:	If this parameter is not provided, the mirror server is determined by querying the SQL Server that is hosting the Site Configuration Database.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input into this cmdlet.

Outputs

Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Site

This includes the name of the Site, the Delivery Controllers in Site, the Databases supporting the Site, the metadata associated with the Site, the licensing configuration of the Site, and the default Icon for the Site.

Notes

The parameter set used with this cmdlet is dependent on the manner in which the databases were previously created.

If the databases were created with default names on a single SQL Server, use the following parameter set:

New-XDSite -AllDefaultDatabases -DatabaseServer <String> -SiteName <String> [-DatabaseMirrorServer <String>] [-DatabaseNamePrefix <String>] [-AdminAddress <String>]

If the databases were created with non-default names on a single SQL Server, use the following parameter set:

New-XDSite -DatabaseServer <String> -LoggingDatabaseName <String> -MonitorDatabaseName <String> -SiteDatabaseName <String> -SiteName <String> [-DatabaseMirrorServer <String>] [-AdminAddress <String>]

If the databases were created on seperate SQL Servers, use the following parameter set:

New-XDSite -LoggingDatabaseName <String> -LoggingDatabaseServer <String> -MonitorDatabaseName <String> -MonitorDatabaseServer <String> -SiteDatabaseName <String> -SiteDatabaseServer <String> -SiteName <String> [-LoggingDatabaseMirrorServer <String>] [-MonitorDatabaseMirrorServer <String>] [-SiteDatabaseMirrorServer <String>] [-AdminAddress <String>]

The command can fail for the following reasons:

- The Site associated with the Controller at AdminAddress is already configured.
- One or more of the specified databases could not be found.
- One or more of the specified databases does not have the necessary permissions for the Controller at AdminAddress.

Related Links

- about_XenDesktopModule_SiteConfiguration
- Get-XDDatabaseSchema

- New-XDDatabase
- Set-XDLicensing
- Get-XDSite

Remove-XDController

March 11, 2024

Removes a Delivery Controller from an existing Site

Syntax

```
Remove-XDController
      -ControllerName <String>
      [-DoNotUpdateDatabaseServer]
      [-AdminAddress <String>]
      [<CommonParameters>]
```

```
Remove-XDController
      -ControllerName <String>
      -DatabaseCredentials <PSCredential>
      [-AdminAddress <String>]
      [<CommonParameters>]
```

```
Remove-XDController
      -ControllerName <String>
      -LoggingDatabaseCredentials <PSCredential>
      -MonitorDatabaseCredentials <PSCredential>
      -SiteDatabaseCredentials <PSCredential>
      [-AdminAddress <String>]
      [<CommonParameters>]
```

Description

Removes a Controller from an existing Site that currently has more than one Controller.

If no credentials are provided and DoNotUpdateDatabaseServer has not been specified, then all database update operations are attempted using the credentials of the user running the cmdlet.

If DatabaseCredentials is provided, then all database operations are attempted using the credentials.

If one or more of SiteDatabaseCredentials, LoggingDatabaseCredentials or MonitorDatabaseCredentials are provided, then all three must be provided. All operations are then attempted using the credentials appropriate to each Database.

Examples

EXAMPLE 1

Removes the Controller 'MyControllerToRemove'from the Site to which 'MyExistingController'is joined. The permissions associated with 'MyControllerToRemove'will be automatically removed from the Database using the implicit credentials of the user running the cmdlet.

```
Remove-XDController -AdminAddress MyExistingController -ControllerName
    MyControllerToRemove
```

EXAMPLE 2

Removes the Controller 'MyControllerToRemove'from the Site to which 'MyExistingController'is joined.

The permissions for 'MyNewController'will not be removed from the Database. Instead the Get-XDDatabaseSchema cmdlet is used to generate a SQL script that may be given to the database administrator of 'MySQLServer'. The SQL script may be used to remove the permissions that previously allowed 'MyControllerToRemove'to manipulate data within the Database.

```
Remove-XDController -AdminAddress MyExistingController -ControllerName
    MyControllerToRemove -DoNotUpdateDatabaseServer
Get-XDDatabaseSchema -AdminAddress MyExistingController -SiteName
    MySite -DatabaseServer MySQLServer -ScriptType RemoveController -
    ControllerToRemoveSid MyControllerToRemove_SID > C:\removeController
    .sql
# Although not necessary, Citrix recommends you get a database
    administrator to run removeController.sql
```

EXAMPLE 3

Removes the Controller 'MyControllerToRemove'from the Site to which 'MyExistingController'is joined.

The credentials contained within '$credential'will be used to remove the permissions associated with 'MyControllerToRemove'from the Database.

```
$credential = Get-Credential
```



```
Remove-XDController -AdminAddress MyExistingController -ControllerName
    MyControllerToRemove -DatabaseCredentials $credential
```

EXAMPLE 4

Removes the Controller 'MyControllerToRemove'from the Site to which 'MyExistingController'is joined.

The credentials contained within '$credentialSite'will be used to remove the permissions associated with 'MyControllerToRemove'from the Site Configuration Database. The credentials contained within '$credentialLogging'will be used to remove the permissions associated with 'MyControllerToRemove' from the Configuration Logging Database. The credentials contained within '$credentialMonitor'will be used to remove the permissions associated with 'MyControllerToRemove'from the Monitoring Database.

```
$credentialSite = Get-Credential
$credentialLogging = Get-Credential
$credentialMonitor = Get-Credential
Remove-XDController -AdminAddress MyExistingController -ControllerName
    MyControllerToRemove -SiteDatabaseCredentials $credentialSite -
    LoggingDatabaseCredentials $credentialLogging -
    MonitorDatabaseCredentials $credentialMonitor
```

Parameters

-ControllerName

The name of the Controller that is to be removed. This may be in SID, DNS or SAM name form. This cannot be the same as the Controller identified by AdminAddress.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Servers associated with the Site Configuration, Configuration Logging and Monitoring Databases.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-LoggingDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Configuration Logging Database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-MonitorDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Monitoring Database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True

Accept pipeline input:	False
Accept wildcard characters:	False

-SiteDatabaseCredentials

The security credentials, in the form of a PSCredential object, that will be used to connect to the SQL Server associated with the Site Configuration Database.

Type:	PSCredential
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DoNotUpdateDatabaseServer

Results in the permissions associated with the Controller not being automatically removed from the Database.

This switch may be useful when the person running the cmdlet has insufficient database privilege. The database permissions for the Controller must be removed manually after running this cmdlet.

Get-XDDatabaseSchema may be used to generate the SQL script which in turn may be used to remove the associated permissions from the Database.

This parameter results in all supplied credentials being ignored.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Notes

If the user running this cmdlet does not have privilege to remove users from the Database, then either use the DoNotudpateDatabaseServer switch or supply alternative credentials using one or more of the Credentials arguments.

This cmdlet cannot be used to remove the last Controller from a Site; use the Remove-XDSite cmdlet instead.

The command can fail for the following reasons:

- The Site has only one Controller. A Site must have multiple Controllers before a Controller can be removed.
- The Controller that is to be removed is not a member of the Site.
- The person running the cmdlet, or the supplied credentials, does not have sufficient privilege to update the Database. Either run this cmdlet as a user that has administrator privilege to the Database, or supply an account with suitable privileges using the PSCredential parameter(s).

Related Links

- Add-XDController
- Get-XDDatabaseSchema
- Get-XDSite
- Remove-XDSite

Remove-XDSite

March 11, 2024

Removes the XenDesktop Site.

Syntax

```
Remove-XDSite
       [-AdminAddress <String>]
       [<CommonParameters>]
```

Description

This cmdlet can be executed only when there is one remaining Controller in the Site. It removes the Site by removing the last Controller from the Site.

The person running this command must be a the local administrator on the last Controller in this Site.

Examples

Parameters

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None

By default, this cmdlet returns no output.

Related Links

- New-XDSite
- Add-XDController
- Remove-XDController

Set-XDLicensing

March 11, 2024

Changes one or more of the licensing attributes of a Site

Syntax

```
Set-XDLicensing
   [-Force]
   [-LicenseServerAddress <String>]
   [-LicenseServerPort <String>]
   [-LicensingModel <String>]
   [-PassThru]
   [-ProductCode <String>]
   [-ProductEdition <String>]
   [-AdminAddress <String>]
   [<CommonParameters>]
```

Description

Changes one or more of the licensing attributes of the Site which has a Controller identified by AdminAddress. When the Citrix License Server address and/or port are specified and Force is not, the new server:port combination will be validated before being changed. Otherwise, no validation is performed.

Examples

EXAMPLE 1

For the Site managed by MyController, changes the License Server to MyLicenseServer:27001.

```
Set-XDLicensing -AdminAddress MyController -LicenseServerAddress
   MyLicenseServer -LicenseServerPort 27001
```


Parameters

-Force

Force the setting of the License Server address and/or port, i.e. do not check that the new server:port combination is valid. This can be used, for example, when the License Server needs to be set but is temporarily unavailable.

Type:	SwitchParameter
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseServerAddress

The address of the License Server for this Site. If this parameter is not provided then the License Server address will not be altered.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicenseServerPort

The port on which the License Server for this Site is listening. If this parameter is not provided then the License Server port will not be altered.

Type:	String
Position:	Named

Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-LicensingModel

The licensing model for this Site. If this parameter is not provided then the licensing model will not be altered. Use the ProductCode returned from Get-XDSite as input to Get-ConfigLicensingModel to determine the list of valid licensing models.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductCode

The product code for this Site. If this parameter is not provided then the product code will not be altered. Find valid product codes by running Get-ConfigProduct.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-ProductEdition

The product edition for this Site. If this parameter is not provided then the product edition will not be altered. Use the ProductCode returned from Get-XDSite as input to Get-ConfigProductEdition to determine the list of valid product editions.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None or Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.LicenseInformation

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.LicenseInformation object.

Notes

The command can fail for the following reasons:

- The supplied License Server does not support the requested product edition.
- A License Server was not found at the supplied address.

Related Links

- Get-ConfigLicensingModel
- Get-ConfigProductEdition
- Get-XDLicensing
- Get-XDSite

Set-XDLogging

March 11, 2024

Sets the Configuration Logging Database attributes of a Site.

Syntax

```
Set-XDLogging
   [-DatabaseMirrorServer <String>]
   [-DatabaseName <String>]
   [-DatabaseServer <String>]
   [-Enabled <Boolean>]
   [-Locale <String>]
   [-PassThru]
   [-AdminAddress <String>]
   [<CommonParameters>]
```

Description

Sets the Configuration Logging Database attributes of the Site which has a Controller identified by AdminAddress.

Examples

EXAMPLE 1

For the Site managed by MyController, enables Configuration Logging.

```
Set-XDLogging -AdminAddress MyController -Enabled $true
```

EXAMPLE 2

For the Site managed by MyController, enables Configuration Logging and allows configuration operations to succeed even if the Configuration Logging Database is unavailable. Note that Configuration Logging must be enabled before AllowDisconnectedDatabase can be set to $true.

```
Set-XDLogging -AdminAddress MyController -Enabled $true -
   AllowDisconnectedDatabase $true
```

EXAMPLE 3

For the Site managed by MyController, sets the mirror for the Configuration Logging Database to MySQLMirror.


```
Set-XDLogging -AdminAddress MyController -DatabaseMirrorServer
    MySQLMirror
```

EXAMPLE 4

For the Site managed by MyController, sets the SQL Server for the Configuration Logging Database to MySQLServer and the database name to MyConfigLoggingDatabase.

```
Set-XDLogging -AdminAddress MyController -DatabaseServer MySQLServer -
    DatabaseName MyConfigLoggingDatabase
```

Parameters

-DatabaseMirrorServer

The mirror database server for the Configuration Logging Database. If this parameter is not provided, the mirror server is determined by querying the SQL Server hosting the Configuration Logging Database.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

The name of the Configuration Logging Database. If this is omitted, the name is assumed to be the existing Configuration Logging Database name.

Type:	String
Position:	Named
Default value:	None
Required:	False

Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseServer

The database server for the Configuration Logging Database. If not provided, the server is assumed to be the SQL Server that is hosting the existing Configuration Logging Database.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Enabled

Determines if Configuration Logging is enabled.

Type:	Boolean
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-Locale

The language used for Configuration Logging entries. Supported locales are English, Japanese and Chinese.

Type:	String

Accepted values:	de, en, es, fr, ja, zh-CN
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None or Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Logging

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Logging object.

Notes

When Configuration Logging is enabled, the dynamic parameter "AllowDisconnectedDatabase"may be used to permit Configuration operations to proceed even if the Configuration Logging Database is not available.

The command can fail for the following reasons:

- The Configuration Logging Database name and/or database server have not been provided, yet there is no existing database name and/or database server to use as default settings.
- The Controller identfied by AdminAddress does not have the necessary permissions to access the new Configuration Loggging Database or mirror database on the database server.
- The new Configuration Loggging Database or mirror database on the database server is not configured for the Site that is associated with the Controller identified by AdminAddress.

Related Links

- Get-XDLogging
- Get-XDSite

Set-XDMonitor

March 11, 2024

Sets the Monitoring Database attributes of a Site.

Syntax

```
Set-XDMonitor
    [-DatabaseMirrorServer <String>]
    [-DatabaseName <String>]
    [-DatabaseServer <String>]
    [-PassThru]
    [-AdminAddress <String>]
    [<CommonParameters>]
```

Description

Sets the Monitoring Database attributes of the Site which has a Controller identified by AdminAddress.

Examples

EXAMPLE 1

For the Site managed by MyController, sets the mirror for the Monitoring Database to MySQLMirror.

```
Set-XDMonitor -AdminAddress MyController -DatabaseMirrorServer
    MySQLMirror
```

EXAMPLE 2

For the Site managed by MyController, sets the SQL Server for the Monitoring Database to MySQLServer and the database name to MyMonitorDatabase.

```
Set-XDMonitor -AdminAddress MyController -DatabaseServer MySQLServer -
    DatabaseName MyMonitorDatabase
```


Parameters

-DatabaseMirrorServer

The mirror database server for the Monitoring Database. If this parameter is not provided, the mirror server is determined by querying the SQL Server hosting the Monitoring Database.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseName

The name of the Monitoring Database. If this is omitted, the name is assumed to be the existing Monitoring Database name.

Type:	String
Position:	Named
Default value:	None
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-DatabaseServer

The database server for the Monitoring Database. If not provided, the server is assumed to be the SQL Server that is hosting the existing Monitoring Database.

Type:	String
Position:	Named
Default value:	None

Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

You cannot pipe input to this cmdlet.

Outputs

None or Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Monitor

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Monitor object.

Notes

The command can fail for the following reasons:

- The Monitoring Database name and/or database server have not been provided, yet there is no existing database name and/or database server to use as default settings.
- The Controller identfied by AdminAddress does not have the necessary permissions to access the new Monitoring Database or mirror database on the database server.
- The new Monitoring Database or mirror database on the database server is not configured for the Site that is associated with the Controller identified by AdminAddress.

Related Links

- Get-XDMonitor
- Get-XDSite

Set-XDSiteMetadata

March 11, 2024

Creates or updates metadata key-value pairs for a Site

Syntax

```
Set-XDSiteMetadata
   -DataName <String>
   -DataValue <String>
   [-PassThru]
   [-AdminAddress <String>]
   [<CommonParameters>]
```

Description

Creates or updates metadata key-value pairs for a Site which has a Controller identified by AdminAddress.

Examples

EXAMPLE 1

Creates, or if it already exists, updates the metadata key “MyMetadataName”with the value “1234”.

```
Set-XDSiteMetadata -DataName "MyMetadataName" -DataValue "1234"
```

Parameters

-DataName

The name of the metadata member that is to be created/updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-DataValue

The value of the metadata member that is to be created/updated.

Type:	String
Position:	Named
Default value:	None
Required:	True
Accept pipeline input:	True (ByPropertyName)
Accept wildcard characters:	True

-PassThru

Returns the affected record. By default, this cmdlet does not generate any output.

Type:	SwitchParameter
Position:	Named
Default value:	False
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

-AdminAddress

Specifies the address of the Delivery Controller to which the PowerShell module will connect. This can be provided as a host name or an IP address.

Type:	String
Position:	Named
Default value:	Localhost. Once a value is provided by any cmdlet, this value will become the default.
Required:	False
Accept pipeline input:	False
Accept wildcard characters:	False

CommonParameters

This cmdlet supports the common parameters: -Debug, -ErrorAction, -ErrorVariable, -InformationAction, -InformationVariable, -OutVariable, -OutBuffer, -PipelineVariable, -Verbose, -WarningAction, and -WarningVariable. For more information, see about_CommonParameters.

Inputs

None

The DataName and DataValue parameters may be piped into this cmdlet by property name.

Outputs

None or Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Site

This cmdlet does not generate any output, unless you use the PassThru parameter, in which case it generates a Citrix.XenDesktopPowerShellSdk.ServiceInterfaces.Configuration.Site object.

Related Links

- Get-XDSite