

Quest® Migration Manager for Active Directory  
8.15

## **Resource Updating Toolkit for**



# PowerShell Reference

**© 2020 Quest Software Inc. ALL RIGHTS RESERVED.**

This guide contains proprietary information protected by copyright. The software described in this guide is furnished under a software license or nondisclosure agreement. This software may be used or copied only in accordance with the terms of the applicable agreement. No part of this guide may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of Quest Software Inc.

The information in this document is provided in connection with Quest Software products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Quest Software products. EXCEPT AS SET FORTH IN THE TERMS AND CONDITIONS AS SPECIFIED IN THE LICENSE AGREEMENT FOR THIS PRODUCT, QUEST SOFTWARE ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL QUEST SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF QUEST SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Quest Software makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Quest Software does not make any commitment to update the information contained in this document.

If you have any questions regarding your potential use of this material, contact:

Quest Software Inc.

Attn: LEGAL Dept

4 Polaris Way

Aliso Viejo, CA 92656

Refer to our Web site (<https://www.quest.com>) for regional and international office information.


**Patents**


Quest Software is proud of our advanced technology. Patents and pending patents may apply to this product. For the most current information about applicable patents for this product, please visit our website at <https://www.quest.com/legal>.

**Trademarks**

Quest, the Quest logo, and Join the Innovation are trademarks and registered trademarks of Quest Software Inc. For a complete list of Quest marks, visit <https://www.quest.com/legal/trademark-information.aspx>. All other trademarks and registered trademarks are property of their respective owners.

**Legend**

 **CAUTION: A CAUTION icon indicates potential damage to hardware or loss of data if instructions are not followed.**

 **IMPORTANT, NOTE, TIP, MOBILE, or VIDEO:** An information icon indicates supporting information.

Migration Manager Resource Updating Toolkit for PowerShell Reference

Updated - March 2020

Version - 8.15

# Contents

<b>Getting Started</b> .....	<b>9</b>
Using Resource Updating Toolkit for PowerShell Remotely .....	9
Using Resource Updating Toolkit for PowerShell Locally .....	10
<b>Cmdlet Reference</b> .....	<b>11</b>
Add-RumComputer .....	12
Detailed Description .....	12
Syntax .....	12
Parameters .....	12
Inputs .....	14
Outputs .....	14
Examples .....	14
Add-RumDomainCredential .....	15
Detailed Description .....	15
Syntax .....	15
Parameters .....	15
Inputs .....	17
Outputs .....	17
Examples .....	17
Copy-RumTask .....	17
Detailed Description .....	17
Syntax .....	17
Parameters .....	18
Inputs .....	21
Outputs .....	21
Examples .....	21
Get-RumCollection .....	21
Detailed Description .....	21
Syntax .....	21
Parameters .....	21
Inputs .....	22
Outputs .....	22
Examples .....	22
Get-RumComputer .....	23
Detailed Description .....	23
Syntax .....	23
Parameters .....	23
Inputs .....	24
Outputs .....	24
Examples .....	25
Get-RumConfiguration .....	25
Detailed Description .....	25

Syntax .....	25
Parameters .....	25
Inputs .....	25
Outputs .....	25
Examples .....	26
Get-RumDomainCredential .....	26
Detailed Description .....	26
Syntax .....	26
Parameters .....	26
Inputs .....	27
Outputs .....	27
Examples .....	28
Get-RumProject .....	28
Detailed Description .....	28
Syntax .....	28
Parameters .....	28
Inputs .....	28
Outputs .....	28
Examples .....	28
Get-RumResult .....	29
Detailed Description .....	29
Syntax .....	29
Parameters .....	29
Inputs .....	30
Outputs .....	30
Examples .....	30
Get-RumTask .....	30
Detailed Description .....	30
Syntax .....	30
Parameters .....	31
Inputs .....	32
Outputs .....	32
Examples .....	32
New-RumCleanupTask .....	32
Detailed Description .....	32
Syntax .....	33
Parameters .....	33
Inputs .....	35
Outputs .....	35
Examples .....	36
New-RumCollection .....	36
Detailed Description .....	36
Syntax .....	36
Parameters .....	36
Inputs .....	37
Outputs .....	37
Examples .....	37

New-RumDiscoveryTask .....	37
Detailed Description .....	37
Syntax .....	38
Parameters .....	38
Inputs .....	40
Outputs .....	40
Examples .....	41
New-RumMoveTask .....	41
Detailed Description .....	41
Syntax .....	41
Parameters .....	41
Inputs .....	47
Outputs .....	47
Examples .....	47
New-RumProcessingTask .....	47
Detailed Description .....	48
Syntax .....	48
Parameters .....	48
Inputs .....	52
Outputs .....	52
Examples .....	52
New-RumRenameTask .....	52
Detailed Description .....	52
Syntax .....	53
Parameters .....	53
Inputs .....	58
Outputs .....	58
Examples .....	58
New-RumScriptingTask .....	58
Detailed Description .....	58
Syntax .....	58
Parameters .....	58
Inputs .....	60
Outputs .....	61
Examples .....	61
Remove-RumCollection .....	61
Detailed Description .....	61
Syntax .....	61
Parameters .....	61
Inputs .....	62
Outputs .....	62
Examples .....	62
Remove-RumComputer .....	63
Detailed Description .....	63
Syntax .....	63
Parameters .....	63
Inputs .....	65

Outputs .....	65
Examples .....	65
Remove-RumDomainCredential .....	65
Detailed Description .....	65
Syntax .....	65
Parameters .....	66
Inputs .....	67
Outputs .....	67
Examples .....	67
Reset-RumCollectionDataProvider .....	67
Detailed Description .....	67
Syntax .....	68
Parameters .....	68
Inputs .....	68
Outputs .....	68
Examples .....	69
Set-RumCollectionDataProvider .....	69
Detailed Description .....	69
Syntax .....	69
Parameters .....	69
Inputs .....	70
Outputs .....	70
Examples .....	71
Set-RumConfiguration .....	71
Detailed Description .....	72
Syntax .....	72
Parameters .....	72
Inputs .....	73
Outputs .....	73
Examples .....	73
Start-RumTask .....	73
Detailed Description .....	74
Syntax .....	74
Parameters .....	74
Inputs .....	75
Outputs .....	75
Examples .....	75
Stop-RumTask .....	75
Detailed Description .....	75
Syntax .....	75
Parameters .....	75
Inputs .....	76
Outputs .....	76
Examples .....	76
Wait-RumUpdate .....	76
Detailed Description .....	76
Syntax .....	77

Parameters .....	77
Inputs .....	79
Outputs .....	79
Examples .....	79
<b>About us .....</b>	<b>80</b>
Technical support resources .....	80



# Getting Started

Resource Updating Manager operation can be automated using Resource Updating Toolkit for PowerShell (PowerRUM). This is a collection of Windows PowerShell cmdlets that replicate the functionality of the Resource Updating Manager console: you can use them to create, view and remove collections, computers and tasks, run tasks, and so on. These command-line tools are useful whenever your resource processing scenarios lend themselves to automation through scripts.

The following software is required on the computer where you run Resource Updating Toolkit cmdlets:

- Windows Management Framework 3.0
- One of the following:
  - Migration Manager console
  - Standalone Resource Updating Manager console

You can work locally on this computer or connect to it remotely.

## Using Resource Updating Toolkit for PowerShell Remotely

On the remote computer, run the PowerShell prompt, and start your session as follows (changing the path and target computer as appropriate):

```
Enable-PSRemoting -Force

$cred = Get-Credential

Enter-PSSession -ComputerName COMPUTER -Credential $cred -ConfigurationName
Microsoft.PowerShell132

Set-Executionpolicy -ExecutionPolicy Unrestricted -Force

cd "C:\Program Files (x86)\Quest Software\Migration Manager\PowerRUM"

Import-Module .\PowerRUM.dll
```

# Using Resource Updating Toolkit for PowerShell Locally

Run the 32-bit (x86) version of the PowerShell prompt, and start your session as follows (changing the path if necessary):

```
Set-Executionpolicy -ExecutionPolicy Unrestricted -Force  
cd "C:\Program Files (x86)\Quest Software\Migration Manager\PowerRUM"  
Import-Module .\PowerRUM.dll
```

# Cmdlet Reference

The following list contains links to the help topics for Resource Updating Toolkit for PowerShell cmdlets.

Name	Description
<a href="#">Add-RumComputer</a>	Adds a computer to a collection
<a href="#">Add-RumDomainCredential</a>	Adds an domain credential to a collection or project root; the credential will be used for access to the specified domain
<a href="#">Copy-RumTask</a>	Makes a copy of an existing task
<a href="#">Get-RumCollection</a>	Returns existing collections
<a href="#">Get-RumComputer</a>	Returns existing computers in a collection
<a href="#">Get-RumConfiguration</a>	Returns the current configuration
<a href="#">Get-RumDomainCredential</a>	Returns the objects representing domain credential
<a href="#">Get-RumProject</a>	Returns a list of existing projects
<a href="#">Get-RumResult</a>	Returns task results
<a href="#">Get-RumTask</a>	Returns existing tasks for the specified collection
<a href="#">New-RumCleanupTask</a>	Creates a new cleanup task
<a href="#">New-RumCollection</a>	Creates a new collection in current RUM project
<a href="#">New-RumDiscoveryTask</a>	Creates a new discovery task
<a href="#">New-RumMoveTask</a>	Creates a new move task
<a href="#">New-RumProcessingTask</a>	Creates a new processing task
<a href="#">New-RumRenameTask</a>	Creates a new rename task
<a href="#">New-RumScriptingTask</a>	Creates a new scripting task
<a href="#">Remove-RumCollection</a>	Removes a collection from the project
<a href="#">Remove-RumComputer</a>	Removes a computer from a collection
<a href="#">Remove-RumDomainCredential</a>	Removes account-representing objects from a collection
<a href="#">Reset-RumCollectionDataProvider</a>	Resets currently specified external account mapping data provider for a collection
<a href="#">Set-RumCollectionDataProvider</a>	Sets an external account mapping data provider for a collection
<a href="#">Set-RumConfiguration</a>	Modifies configuration parameters

Name	Description
<a href="#">Start-RumTask</a>	Runs a task
<a href="#">Stop-RumTask</a>	Stops a running task
<a href="#">Wait-RumUpdate</a>	Waits until tasks or processing of computers complete with specified statuses.

# Add-RumComputer

Adds a computer to a collection.

## Detailed Description

The **Add-RumComputer** cmdlet adds a computer to a collection; the computer is not necessarily included in the project.

## Syntax

```
Add-RumComputer [-Computer] <PSRumComputer[]>
-CollectionName <String> [-PassThru] [<CommonParameters>]

Add-RumComputer [-Computer] <PSRumComputer[]> [
-Collection] <PSRumCollection> [-PassThru] [<CommonParameters>]

Add-RumComputer [-ComputerName] <String[]>
-CollectionName <String> [-ComputerDomain <String>] [
-PassThru] [<CommonParameters>]

Add-RumComputer [-ComputerName] <String[]> [
-Collection] <PSRumCollection> [-ComputerDomain <String>] [
-PassThru] [<CommonParameters>]
```

## Parameters

**-Computer** <PSRumComputer[]>

The computer to add to the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ComputerName** <String[]>

The name of the computer to add to the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Collection <PSRumCollection>**

The collection to add the computer to.

Required?	<b>true</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

The name of the collection to add the computer to

Required?	<b>true</b>
Position?	<b>named</b>
Default value	
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ComputerDomain <String>**

The domain of the computer to add to the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **-PassThru**

Returns the objects that represent the collections that were changed. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## **Inputs**

PSRumCollection  
PSRumComputer  
String

## **Outputs**

PSRumComputer

## **Examples**

### **Example 1**

```
Get-RumCollection "collection_1" | Add-RumComputer  
"domain_1\computer_1"
```

### **Example 2**

```
Get-RumCollection "collection_1" | Add-RumComputer "computer_1"  
-ComputerDomain "domain_1"
```

### **Example 3**

```
"computer_1", "computer_2", "computer_3" | Add-RumComputer  
-CollectionName "collection_1" -ComputerDomain "domain_1"
```

## Example 4

```
Get-RumComputer -CollectionName "collection_1" | Add-RumComputer  
-CollectionName "collection_2"
```

# Add-RumDomainCredential

Adds an domain credential to a collection or project root; the credential will be used for access to the specified domain.

## Detailed Description

The **Add-RumDomainCredential** cmdlet adds an domain credential to a collection or project root; the credential will be used for access to the specified domain.

## Syntax

```
Add-RumDomainCredential [-Name] <String> [  
-Credential] <PSCredential> [[-Collection] <PSRumCollection>] [  
-PassThru] [<CommonParameters>]
```

```
Add-RumDomainCredential [-Name] <String> [  
-Credential] <PSCredential> -CollectionName <String> [  
-PassThru] [<CommonParameters>]
```

## Parameters

**-Name <String>**

Domain.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Credential <PSCredential>**

Credentials for access to the domain.

Required?	<b>true</b>
Position?	<b>2</b>

Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Collection <PSRumCollection>**

Collection.

Required?	<b>false</b>
Position?	<b>3</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represent the domain credential. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).



## Inputs

PSRunCollection

## Outputs

PSRunDomainCredential

## Examples

### Example 1

```
Add-RunDomainCredential "domain_1" (Get-Credential)
```

### Example 2

```
Get-RunCollection "collection_1" | Add-RunDomainCredential  
"domain_1" (Get-Credential)
```

# Copy-RunTask

Makes a copy of an existing task.

## Detailed Description

The **Copy-RunTask** cmdlet makes a copy of an existing task and lets you specify any of the following properties: task name, list of computers to include and when to run the task.

## Syntax

```
Copy-RunTask [-Task] <PSRunTask> [-Collection] <PSRunCollection> [  
-Computer <PSRunComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

```
Copy-RunTask [-TaskId] <Guid> [-Collection] <PSRunCollection> [  
-Computer <PSRunComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

```
Copy-RunTask [-Task] <PSRunTask> -CollectionName <String> [  
-Computer <PSRunComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

```
Copy-RumTask [-TaskId] <Guid> -CollectionName <String> [
-Computer <PSRumComputer[]>] [-Description <String>] [
-InProgressTimeout <Int32>] [-Name <String>] [
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [
<CommonParameters>]
```

## Parameters

### **-Task <PSRumTask>**

Original task.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **-TaskId <Guid>**

Original task ID.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

### **-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

### **-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRumComputer[]>**

List of computers.

Required?	<b>false</b>
Position?	<b>named</b>
Defaultvalue	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the new task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the *In Progress* state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Name of the new task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represent the domain credential. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
PSRunTask

## Outputs

PSRunTask

## Examples

### Example 1

```
Get-RunCollection "collection_1" | Copy-RunTask  
-TaskId $sourceTaskId -PassThru
```

### Example 2

```
Get-RunTask $sourceTaskId | Copy-RunTask -CollectionName "collection_1"  
-PassThru
```

# Get-RunCollection

Returns existing collections.

## Detailed Description

The **Get-RunCollection** cmdlet returns all existing collections that match the specified name wildcard or ID.

## Syntax

```
Get-RunCollection [-Id] <Guid[]> [<CommonParameters>]  
Get-RunCollection [[-Name] <String[]>] [<CommonParameters>]
```

## Parameters

**-Id** <Guid[]>

ID of the requested collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name** <String[]>

Name of the collection.

Required?	<b>false</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>true</b>

#### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

String

Guid

## Outputs

PSRunCollection

## Examples

### Example 1

```
Get-RumCollection
```

### Example 2

```
Get-RumCollection "collection_1"
```

### Example 3

```
Get-RumCollection ([guid]$id)
```

# Get-RumComputer

Returns existing computers in a collection.

## Detailed Description

The **Get-RumComputer** cmdlet returns all existing computers in collection or category that match the specified name wildcard or ID

## Syntax

```
Get-RumComputer [[-Collection] <PSRumCollection>] [  
<CommonParameters>]
```

```
Get-RumComputer [-Id] <Guid[]> [<CommonParameters>]
```

```
Get-RumComputer [[-Name] <String[]>] -CollectionName <String> [  
<CommonParameters>]
```

```
Get-RumComputer [-Name] <String[]> [  
-Collection] <PSRumCollection>] [<CommonParameters>]
```

## Parameters

**-Id** <Guid[]>

ID of the requested computer.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name** <String[]>

Wildcard for the names of the requested computers.

Required?	<b>false</b>
Position?	<b>1</b>
Default value	<b>none</b>

Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>true</b>

**-Collection <PSRumCollection>**

Collection.

Required?	<b>false</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRumCollection

## Outputs

PSRumComputer



## Examples

### Example 1

```
Get-RumComputer
```

### Example 2

```
Get-RumCollection "collection_1" | Get-RumComputer"
```

### Example 3

```
Get-RumCollection "collection_1" | Get-RumComputer  
"domain_1\computer_1"
```

# Get-RumConfiguration

Returns the current configuration.

## Detailed Description

The **Get-RumConfiguration** cmdlet returns the current Resource Updating Manager configuration.

## Syntax

```
Get-RumConfiguration [<CommonParameters>]
```

## Parameters

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

None

## Outputs

PSRumConfiguration

Object containing the current configuration

## Examples

### Example 1

```
Get-RumConfiguration
```

# Get-RumDomainCredential

Returns the objects representing domain credential.

## Detailed Description

The **Get-RumDomainCredential** cmdlet returns the objects representing domain credential in the specified collection or project root. The objects can be requested by wildcard or by ID.

## Syntax

```
Get-RumDomainCredential [[-Collection] <PSRumCollection>] [  
<CommonParameters>]
```

```
Get-RumDomainCredential [-Id] <Guid[]> [<CommonParameters>]
```

```
Get-RumDomainCredential [[-Name] <String[]>]  
-CollectionName <String> [<CommonParameters>]
```

```
Get-RumDomainCredential [-Name] <String[]> [[  
-Collection] <PSRumCollection>] [<CommonParameters>]
```

## Parameters

**-Id** <Guid[]>

ID of the requested object.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name** <String[]>

Wildcard of domain name for requested objects.

Required?	<b>false</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>true</b>

**-Collection <PSRumCollection>**

Collection.

Required?	<b>false</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRumCollection

## Outputs

PSRumDomainCredential

## Examples

### Example 1

```
Get-RumDomainCredential "domain_1"
```

### Example 2

```
Get-RumCollection "collection_1" | Get-RumDomainCredential  
"domain_1"
```

# Get-RumProject

Returns a list of existing projects.

## Detailed Description

The **Get-RumProject** cmdlet returns a list of existing projects.

## Syntax

```
Get-RumProject [<CommonParameters>]
```

## Parameters

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

None

## Outputs

String

## Examples

### Example 1

```
Get-RumProject
```

# Get-RumResult

Returns task results.

## Detailed Description

The **Get-RumResult** cmdlet returns the results for specified tasks or computers.

## Syntax

```
Get-RumResult [-Computer] <PSRumComputer> [<CommonParameters>]
```

```
Get-RumResult [-Task] <PSRumTask> [<CommonParameters>]
```

```
Get-RumResult [-Computer] <PSRumComputer> [-Task] <PSRumTask> [  
<CommonParameters>]
```

## Parameters

**-Computer** <PSRumComputer>

Computer.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Task** <PSRumTask>

Task.

Required?	<b>true</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunComputer  
PSRunTask

## Outputs

PSRunResult  
Objects containing information about tasks performed

## Examples

### Example 1

```
Get-RunTask -CollectionName "collection_1" | Get-RunResult
```

### Example 2

```
Get-RunComputer -CollectionName "collection_1" "computer_1" |  
Get-RunResult
```

### Example 3

```
Get-RunTask -CollectionName "collection_1" | Get-RunResult  
(Get-RunComputer -CollectionName "collection_1" "computer_1")
```

# Get-RunTask

Returns existing tasks for the specified collection

## Detailed Description

The **Get-RunTask** cmdlet returns existing tasks for the specified collection. You can request all tasks, tasks by wildcard or a task by ID.

## Syntax

```
Get-RunTask [[-Collection] <PSRunCollection>] [<CommonParameters>]
```

```
Get-RunTask [-Id] <Guid[]> [<CommonParameters>]
```

```
Get-RunTask [[-Name] <String[]>] -CollectionName <String> [  
<CommonParameters>]
```

```
Get-RunTask [-Name] <String[]> [[-Collection] <PSRunCollection>] [  
<CommonParameters>]
```

# Parameters

**-Id <Guid[]>**

ID of the requested task.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name <String[]>**

Wildcard for the names of the requested tasks.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>true</b>

**-Collection <PSRunCollection>**

Collection.

Required?	<b>false</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>

Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

---

#### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
Collection

## Outputs

PSRunTask  
Task

## Examples

### Example 1

```
Get-RunTask
```

### Example 2

```
Get-RunCollection "collection_1" | Get-RunTask
```

### Example 3

```
Get-RunTask -CollectionName "collection_1"
```

### Example 4

```
Get-RunTask -Id $taskId
```

# New-RunCleanupTask

Creates a new cleanup task..

## Detailed Description

The **New-RunCleanupTask** cmdlet creates a new cleanup task.



## Syntax

```
New-RumCleanupTask [-Collection] <PSRumCollection> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

```
New-RumCleanupTask -CollectionName <String> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

## Parameters

**-Collection** <PSRumCollection>

Collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName** <String>

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer** <PSRumComputer[]>

Computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>

Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

Collection

## Outputs

PSRunTask

# Examples

## Example 1

```
Get-RumCollection "collection_1" | New-RumCleanupTask  
-Name "Task_Cleanup" -Schedule "2013/12/31 23:59"
```

# New-RumCollection

Creates a new collection in current project.

## Detailed Description

The **New-RumCollection** cmdlet creates a new collection in current RUM project.

## Syntax

```
New-RumCollection [-Name] <String> [-Description <String>] [  
-PassThru] [  
<CommonParameters>]
```

## Parameters

**-Name <String>**

Collection name

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the collection.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **-PassThru**

Returns the object that represents the collection that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## **Inputs**

None

## **Outputs**

PSRunTask

## **Examples**

### **Example 1**

```
New-RumCollection "new collection name"
```

# **New-RumDiscoveryTask**

Creates a new discovery task.

## **Detailed Description**

The **New-RumDiscoveryTask** cmdlet creates a new discovery task.

## Syntax

```
New-RumDiscoveryTask [-Collection] <PSRumCollection> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

```
New-RumDiscoveryTask -CollectionName <String> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

## Parameters

**-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRumComputer[]>**

Collection member computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>

Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
PSRunComputer

## Outputs

PSRunTask



## Examples

### Example 1

```
$task = Get-RumCollection "collection_1" | New-RumDiscoveryTask -PassThru
```

# New-RumMoveTask

Creates a new move task.

## Detailed Description

The **New-RumMoveTask** cmdlet creates a new move task

## Syntax

```
New-RumMoveTask [-Collection] <PSRumCollection>  
-TargetDomain <String> [-Computer <PSRumComputer[]>] [  
-Description <String>] [-GrantLocalAdmin <String[]>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-RestartMessage <String>] [  
-RestartTimeout <Int32>] [-Schedule <DateTime>] [  
-ScriptAfter <String>] [-ScriptBefore <String>] [  
-TargetContainer <String>] [-PassThru] [-PreserveSourceAccount] [  
-Remote] [-Restart] [-RestartForce] [-UpdateLastLoggedDomain] [  
<CommonParameters>]
```

```
New-RumMoveTask -CollectionName <String> -TargetDomain <String> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-GrantLocalAdmin <String[]>] [-InProgressTimeout <Int32>] [  
-Name <String>] [-PendingTimeout <Int32>] [  
-RestartMessage <String>] [-RestartTimeout <Int32>] [  
-Schedule <DateTime>] [-ScriptAfter <String>] [  
-ScriptBefore <String>] [-TargetContainer <String>] [  
-PassThru] [-PreserveSourceAccount] [-Remote] [-Restart] [  
-RestartForce] [-UpdateLastLoggedDomain] [<CommonParameters>]
```

## Parameters

**-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>

Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRunComputer[]>**

Collection member computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-GrantLocalAdmin <String[]>**

The accounts that will be added to the local Administrators group on the computers you are going to move.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-PreserveSourceAccount**

Makes the cmdlet preserve the computer account in the source domain.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-Remote**

Makes the cmdlet perform the task remotely.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-Restart**

Makes the cmdlet restart the computer.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-RestartForce**

Makes the cmdlet force restart.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-RestartMessage <String>**

Restart message.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-RestartTimeout <Int32>**

Restart timeout.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>

Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptAfter <String>**

Script to run after the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptBefore <String>**

Script to run before the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-TargetContainer <String>**

Target container DN.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-TargetDomain <String>**

Target domain.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-UpdateLastLoggedDomain**

Makes the cmdlet change the last logged-in domain to the target domain.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
PSRunComputer

## Outputs

PSRunTask

## Examples

### Example 1

```
$task = Get-RumCollection "collection_1" | New-RumMoveTask
-PassThru -TargetDomain "domain"
```

# New-RumProcessingTask

Creates a new processing task.

## Detailed Description

The **New-RumProcessingTask** cmdlet creates a new processing task.

## Syntax

```
New-RumProcessingTask [-Collection] <PSRumCollection> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Process <ProcessResource>] [  
-ProcessingMode <RumProcessingModes>] [-Schedule <DateTime>] [  
-ScriptAfter <String>] [-ScriptBefore <String>]  
[-SidHistoryTargetDomain <String>] [-PassThru] [  
<CommonParameters>]
```

```
New-RumProcessingTask -CollectionName <String> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Process <ProcessResource>] [  
-ProcessingMode <RumProcessingModes>] [-Schedule <DateTime>] [  
-ScriptAfter <String>] [-ScriptBefore <String>]  
[-SidHistoryTargetDomain <String>] [-PassThru] [  
<CommonParameters>]
```

## Parameters

**-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>



**-Computer <PSRunComputer[]>**

Collection member computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>

Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-SidHistoryTargetDomain <String>**

Target domain where SIDHistory of objects will be used for account matching.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Process<ProcessResource>**

The types of objects whose permissions should be re-assigned to target users.

Values : LocalGroupMembership, UserRights, ServiceAccounts, ScheduledTasks, LocalProfiles, RoamingProfiles, Registry, FileSystem, FileOwnership, Shares, Printers, COMPlus, DCOM, IIS, All

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ProcessingMode <RunProcessingModes>**

Re-Permissioning mode.

Values : Reassign, Copy, Cleanup, Revert

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptAfter <String>**

Script to run after the task

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptBefore** <String>

Script to run before the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
PSRunComputer

## Outputs

PSRunTask

## Examples

### Example 1

```
$task = Get-RumCollection "collection_1" | New-RumProcessingTask  
-PassThru -Process All
```

### Example 2

```
New-RumProcessingTask -Name "task_1" -CollectionName "collection_2"  
-SidHistoryTargetDomain domainName.local -ProcessingMode Copy -Process Shares,Printers
```

# New-RumRenameTask

Creates a new rename task.

## Detailed Description

The **New-RumRenameTask** cmdlet creates a new rename task.

## Syntax

```
New-RumRenameTask [-Collection] <PSRumCollection>  
-RenameFile <String> [-Computer <PSRumComputer[]>] [  
-Description <String>] [-InProgressTimeout <Int32>] [  
-Name <String>] [-PendingTimeout <Int32>] [-Remote <Boolean>] [  
-RestartMessage <String>] [-RestartTimeout <Int32>] [  
-Schedule <DateTime>] [-ScriptAfter <String>] [  
-ScriptBefore <String>] [-PassThru] [-Restart] [-RestartForce] [  
<CommonParameters>]
```

```
New-RumRenameTask -CollectionName <String> -RenameFile <String> [  
-Computer <PSRumComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Remote <Boolean>] [  
-RestartMessage <String>] [-RestartTimeout <Int32>] [  
-Schedule <DateTime>] [-ScriptAfter <String>] [  
-ScriptBefore <String>] [-PassThru] [-Restart] [-RestartForce] [  
<CommonParameters>]
```

## Parameters

**-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRumComputer[]>**

Collection member computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PendingTimeout <Int32>**

The timeout in minutes for keeping the task in the Pending state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Remote <Boolean>**

Makes the cmdlet perform the task remotely.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-RenameFile <String>**

File containing pairs of the old and new computer names.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-Restart**

Makes the cmdlet restart the computer.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-RestartForce**

Makes the cmdlet force restart.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-RestartMessage <String>**

Restart message.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **-RestartTimeout <Int32>**

Restart timeout.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>



Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Schedule <DateTime>**

When to run the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptAfter <String>**

Script to run after the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ScriptBefore <String>**

Script to run before the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection

PSRunComputer

## Outputs

PSRunTask

## Examples

### Example 1

```
$task = Get-RumCollection "collection_1" | New-RumRenameTask  
-PassThru -RenameFile c:\rename.txt
```

# New-RumScriptingTask

Creates a new scripting task.

## Detailed Description

The **New-RumScriptingTask** cmdlet creates a new scripting task.

## Syntax

```
New-RumScriptingTask [-Collection] <PSRunCollection>  
-Script <String> [-Computer <PSRunComputer[]>] [  
-Description <String>] [-InProgressTimeout <Int32>] [  
-Name <String>] [-PendingTimeout <Int32>] [-Schedule <DateTime>] [  
-PassThru] [<CommonParameters>]
```

```
New-RumScriptingTask -CollectionName <String> -Script <String> [  
-Computer <PSRunComputer[]>] [-Description <String>] [  
-InProgressTimeout <Int32>] [-Name <String>] [  
-PendingTimeout <Int32>] [-Schedule <DateTime>] [-PassThru] [  
<CommonParameters>]
```

## Parameters

**-Collection** <PSRunCollection>

Collection.

Required?	<b>true</b>
-----------	-------------

Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of the collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRunComputer[]>**

Collection member computers where the task will run.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Description <String>**

Description of the task.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-InProgressTimeout <Int32>**

The timeout in minutes for keeping the task in the In Progress state.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Name <String>**

Task name.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the task that was created. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection  
PSRunComputer

## Outputs

PSRunTask

## Examples

### Example 1

```
$task = Get-RumCollection "collection_1" | New-RumScriptingTask  
-PassThru -Script c:\script.cmd
```

# Remove-RumCollection

Removes a collection from the project.

## Detailed Description

The **Remove-RumCollection** cmdlet removes a collection from the project

## Syntax

```
Remove-RumCollection [-Collection] <PSRunCollection[]> [  
<CommonParameters>]
```

```
Remove-RumCollection [-Id] <Guid[]> [<CommonParameters>]
```

```
Remove-RumCollection [-Name] <String[]> [<CommonParameters>]
```

## Parameters

**-Collection** <PSRunCollection[]>

Collection to remove.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Id** <Guid[]>

ID of the collection to remove.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name <String[]>**

Name of the collection to remove.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>true</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunCollection

## Outputs

None

## Examples

### Example 1

```
Remove-RumCollection "collection_1"
```

### Example 2

```
Get-RumCollection "collection_1" | Remove-RumCollection
```

# Remove-RumComputer

Removes a computer from a collection.

## Detailed Description

The **Remove-RumComputer** cmdlet Removes a computer from a collection without deleting it from the project..

## Syntax

```
Remove-RumComputer [-Computer] <PSRumComputer[]> [-Collection] <PSRumCollection> [<CommonParameters>]
```

```
Remove-RumComputer [-Computer] <PSRumComputer[]> -CollectionName <String> [<CommonParameters>]
```

```
Remove-RumComputer [-Id] <Guid[]> [-Collection] <PSRumCollection> [<CommonParameters>]
```

```
Remove-RumComputer [-Id] <Guid[]> -CollectionName <String> [<CommonParameters>]
```

```
Remove-RumComputer [-Name] <String[]> [-Collection] <PSRumCollection> [<CommonParameters>]
```

```
Remove-RumComputer [-Name] <String[]> -CollectionName <String> [<CommonParameters>]
```

## Parameters

**-Computer** <PSRumComputer[]>

Computer to remove from the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Id** <Guid[]>

ID of the computer to remove from the collection.

Required?	<b>true</b>
Position?	<b>1</b>

Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name <String[]>**

Name of the computer to remove from the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>true</b>

**-Collection <PSRumCollection>**

Collection.

Required?	<b>true</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).



## Inputs

PSRumCollection

## Outputs

None

## Examples

### Example 1

```
Get-RumCollection "collection_1" | Remove-RumComputer  
"domain_1\computer_1"
```

### Example 2

```
Remove-RumComputer "domain_1\computer_1"  
-CollectionName "collection_1"
```

### Example 3

```
Get-RumComputer -CollectionName "collection_1""domain_1\computer_1" | Remove-  
RumComputer -CollectionName "collection_1"
```

# Remove-RumDomainCredential

Removes account-representing objects from a collection..

## Detailed Description

The **Remove-RumDomainCredential** cmdlet removes account-representing objects from a collection; the accounts are used for access to the specified domain.

## Syntax

```
Remove-RumDomainCredential [-DomainCredential] <PSRumDomainCredential[]> [  
<CommonParameters>]
```

```
Remove-RumDomainCredential [-Id] <Guid[]> [<CommonParameters>]
```

```
Remove-RumDomainCredential [-Name] <String[]> -CollectionName <String> [  
<CommonParameters>]
```

```
Remove-RumDomainCredential [-Name] <String[]> [[-Collection] <PSRumCollection>] [  
<CommonParameters>]
```

# Parameters

**-DomainCredential** <PSRumDomainCredential[]>

Credential object to remove from a collection or project root.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Id** <Guid[]>

ID of credential object to remove from a collection or project root.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Name** <String[]>

Domain name of credential object to remove from collection or project root.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>true</b>

**-Collection** <PSRumCollection>

Collection.

Required?	<b>false</b>
Position?	<b>2</b>
Default value	<b>none</b>

Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-CollectionName <String>**

Name of collection.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunDomainCredential

## Outputs

None

## Examples

### Example 1

```
Get-RumCollection "collection_1" | Get-RumDomainCredential "domain" | Remove-RumDomainCredential
```

# Reset-RumCollectionDataProvider

Resets currently specified external account mapping data provider for a collection.

## Detailed Description

The **Reset-RumCollectionDataProvider** cmdlet resets currently specified external account mapping data provider for a collection. Original account mapping settings used prior to changing the data provider with the

[Set-RumCollectionDataProvider](#) cmdlet will be restored for a collection (either default project map or custom map).

## Syntax

```
Reset-RumCollectionDataProvider [-Id] <Guid> [-PassThru] [<CommonParameters>]
```

## Parameters

**-Id <Guid>**

ID of the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue, ByPropertyName)</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the modified collection. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

Guid

## Outputs

PSRumCollection

## Examples

### Example 1

```
Reset-RumCollectionDataProvider b5ba8569-80b2-42ea-a07d-a4cc1d3ef7cc
```

### Example 2

```
Get-RumCollection "collectionName" | Reset-RumCollectionDataProvider
```

# Set-RumCollectionDataProvider

Sets an external account mapping data provider for a collection.

## Detailed Description

The **Set-RumCollectionDataProvider** cmdlet specifies a data provider from which to obtain account mapping data instead of account map currently utilized for a collection in the project (default project map or custom map).

## Syntax

```
Set-RumCollectionDataProvider [-Id] <Guid> [-DataProviderKey] <String>  
[-DataProviderConfig] <String> [-PassThru] [<CommonParameters>]
```

## Parameters

**-Id <Guid>**

ID of the collection.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue, ByPropertyName)</b>
Accept wildcard characters?	<b>false</b>

**-DataProviderKey <String>**

Key of data provider. Currently, only the account mapping data provider named "PowerShell" is available.

Required?	<b>true</b>
Position?	<b>2</b>

Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-DataProviderConfig <String>**

Configuration of the data provider. For data provider "PowerShell" specify script file or script itself that implements the `Get-DomainPairs` and `Get-ObjectPairs ($DomainPair)` functions. See Example 2 below for details.

Required?	<b>true</b>
Position?	<b>3</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the modified collection. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**<CommonParameters>**

This cmdlet supports the common parameters: `Verbose`, `Debug`, `ErrorAction`, `ErrorVariable`, `WarningAction`, `WarningVariable`, `OutBuffer` and `OutVariable`. For more information, see [about\\_CommonParameters](#).

## Inputs

Guid

## Outputs

PSRumCollection

## Examples

### Example 1

```
Set-RumCollectionDataProvider "b5ba8569-80b2-42ea-a07d-a4ccd3ef7cc"  
-DataProviderKey PowerShell -DataProviderConfig ". C:\script.ps1"
```

### Example 2

```
$tmp = 'function Get-DomainPairs {  
@{ Source = @{SID="S-1-5-21-sourceDomainSID"; Name="source"; DNS="source.local" };  
Target = @{SID="S-1-5-21-targetDomainSID"; Name="target"; DNS="target.local" };  
Id = 1; } }  
  
function Get-ObjectPairs($DomainPair)  
{ $Id = $DomainPair.Get_Item("Id"); switch ( $Id ) { 1 {  
@{ Source = @{SID="S-1-5-21-sourceUserSID"; Name="user.name";  
UPN="user.name@source.local"; };  
Target = @{SID="S-1-5-21-targetUserSID"; Name="user.name";  
UPN="user.name@target.local"; };  
Type="user"; } } default { throw "Unknown item Id: " + $Id; } } }'  
  
Get-RumCollection "collection1" | Set-RumCollectionDataProvider  
-DataProviderKey PowerShell -DataProviderConfig $tmp
```

### Example 3

```
Get-RumCollection "MainCollection" | Set-RumCollectionDataProvider -DataProviderKey  
PowerShell  
-DataProviderConfig '. "$env:CommonProgramFiles\Aelita Shared\Migration Tools\Resource  
Updating\CustomMapFromFile.ps1"  
''\Server\Share\Folder\list.csv''' -PassThru
```

where list.csv file contains a list of migration pairs in the following format, one pair per line:

```
SourceNetBIOSDomainName\SourceUserName,SourceUserSID,SourceUPN,SourceDomainDNSName,Tar  
getNetBIOSDomainName\TargetUserName,TargetUserSID,TargetUPN,TargetDomainDNSName
```

The following is an example of list.csv file:

```
SOURCE1\user1.name,S-1-5-21-  
sourceUser1SID,user1@source1.principal.name,source1.com,TARGET1\user1.name,S-1-5-21-  
targetUser1SID,user1@target1.principal.name,target1.com  
SOURCE2\user2.name,S-1-5-21-  
sourceUser2SID,user2@source2.principal.name,source2.local,TARGET2\user2.name,S-1-5-21-  
targetUser2SID,user2@target2.principal.name,target2.local
```

## Set-RumConfiguration

Modifies configuration parameters.

## Detailed Description

The **Set-RumConfiguration** cmdlet modifies Resource Updating Manager configuration parameters.

## Syntax

```
Set-RumConfiguration -Project <String> [<CommonParameters>]
```

```
Set-RumConfiguration [-Server] <String> [-Port] <Int32> [[  
-Project] <String>] [-ControllerCredential <PSCredential>] [  
<CommonParameters>]
```

## Parameters

**-Server <String>**

AD LDS or ADAM server.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Port <Int32>**

Port number.

Required?	<b>true</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Project <String>**

Project name.

Required?	<b>true</b>
Position?	<b>named</b>
Default value	<b>none</b>



Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-ControllerCredential** <PSCredential>

Credentials for the service controller.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

None

## Outputs

None

## Examples

### Example 1

```
Set-RumConfiguration "server" 389 "CN=QMMADProject"
```

### Example 2

```
Set-RumConfiguration "server" 389; Set-RumConfiguration
-Project (Get-RumProject | select -first 1)
```

# Start-RumTask

Runs a task.

## Detailed Description

The **Start-RunTask** cmdlet runs the task immediately or sets a schedule.

## Syntax

```
Start-RunTask [-Task] <PSRunTask> [[-Schedule] <DateTime>] [-PassThru] [<CommonParameters>]
```

## Parameters

### **-Task** <PSRunTask>

Task.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

### **-Schedule** <DateTime>

When to run the task.

Required?	<b>false</b>
Position?	<b>2</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### **-PassThru**

Returns the object that represents the task. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

### <CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunTask

## Outputs

PSRunTask

## Examples

### Example 1

```
Get-RunTask -CollectionName "collection_1" | Start-RunTask  
-PassThru | Wait-RunUpdate
```

# Stop-RunTask

Stops a running task.

## Detailed Description

The **Stop-RunTask** cmdlet stops a running task

## Syntax

```
Stop-RunTask [-Task] <PSRunTask> [-PassThru] [<CommonParameters>]
```

## Parameters

**-Task** <PSRunTask>

Task.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>

Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

#### **-PassThru**

Returns the object that represents the task. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

#### **<CommonParameters>**

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunTask

## Outputs

PSRunTask

## Examples

### Example 1

```
Get-RumTask "task name" -CollectionName "collection_1" | Stop-RumTask
```

# Wait-RumUpdate

Waits until tasks or processing of computers complete with specified statuses.

## Detailed Description

The **Wait-RumUpdate** cmdlet waits until tasks or processing of computers complete with specified statuses.

## Syntax

```
Wait-RumUpdate [-Task] <PSRumTask[]> [-PollInterval <TimeSpan>] [-Timeout <TimeSpan>] [-WaitTaskStatus <RumTaskStatuses[]>] [-All] [-PassThru] [<CommonParameters>]
```

```
Wait-RumUpdate [-Computer] <PSRumComputer[]> [-PollInterval <TimeSpan>] [-Timeout <TimeSpan>] [-WaitComputerStatus <RumComputerStatuses[]>] [-All] [-PassThru] [<CommonParameters>]
```

## Parameters

**-Task <PSRumTask[]>**

Tasks to be monitored.

Required?	<b>true</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-Computer <PSRumComputer[]>**

Computers to be monitored.

Required?	<b>false</b>
Position?	<b>1</b>
Default value	<b>none</b>
Accept pipeline input?	<b>true (ByValue)</b>
Accept wildcard characters?	<b>false</b>

**-All**

Makes the cmdlet wait until all tasks or processing of all computers complete; otherwise, the waiting stops as soon as any task or processing of any computer completes.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PassThru**

Returns the object that represents the computer or task. By default, this cmdlet does not generate any output.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-PollInterval <TimeSpan>**

Polling interval for status information updating in HH:mm:ss format.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>00:00:10</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-Timeout <TimeSpan>**

Timeout for waiting in HH:mm:ss format. By default, there is no timeout.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>none</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-WaitComputerStatus <RunComputerStatuses []>**

Processing statuses for specified computers that cause cmdlet completion.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>ComputerStatusOk, ComputerStatusFailure, ComputerStatusWarning, ComputerStatusUnknown</b>

Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

**-WaitTaskStatus** <RunTaskStatuses []>

Task statuses that cause cmdlet completion.

Required?	<b>false</b>
Position?	<b>named</b>
Default value	<b>Cancelled, Completed, Failed</b>
Accept pipeline input?	<b>false</b>
Accept wildcard characters?	<b>false</b>

<CommonParameters>

This cmdlet supports the common parameters: Verbose, Debug, ErrorAction, ErrorVariable, WarningAction, WarningVariable, OutBuffer and OutVariable. For more information, see [about\\_CommonParameters](#).

## Inputs

PSRunTask  
PSRunComputer

## Outputs

Object

## Examples

### Example 1

```
Get-RunTask -CollectionName "collection_1" | Start-RunTask -PassThru | Wait-RunUpdate
```

### Example 2

```
New-RunDiscoveryTask -CollectionName $value -PassThru | Start-RunTask -PassThru |  
Wait-RunUpdate -WaitTaskStatus Completed -Timeout 00:01:30
```

# About us

---

Quest provides software solutions for the rapidly-changing world of enterprise IT. We help simplify the challenges caused by data explosion, cloud expansion, hybrid datacenters, security threats, and regulatory requirements. We are a global provider to 130,000 companies across 100 countries, including 95% of the Fortune 500 and 90% of the Global 1000. Since 1987, we have built a portfolio of solutions that now includes database management, data protection, identity and access management, Microsoft platform management, and unified endpoint management. With Quest, organizations spend less time on IT administration and more time on business innovation. For more information, visit [www.quest.com](http://www.quest.com).

## Technical support resources

Technical support is available to Quest customers with a valid maintenance contract and customers who have trial versions. You can access the Quest Support Portal at <https://support.quest.com>.

The Support Portal provides self-help tools you can use to solve problems quickly and independently, 24 hours a day, 365 days a year. The Support Portal enables you to:

- Submit and manage a Service Request
- View Knowledge Base articles
- Sign up for product notifications
- Download software and technical documentation
- View how-to-videos
- Engage in community discussions
- Chat with support engineers online
- View services to assist you with your product