Quest

Metalogix® Replicator 7.4

Command Line Reference Guide

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 Metalogix are trademarks and registered trademarks of Quest Software Inc. and its affiliates. 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.

Metalogix® Replicator
Updated September 2023
Version 7.4

Contents

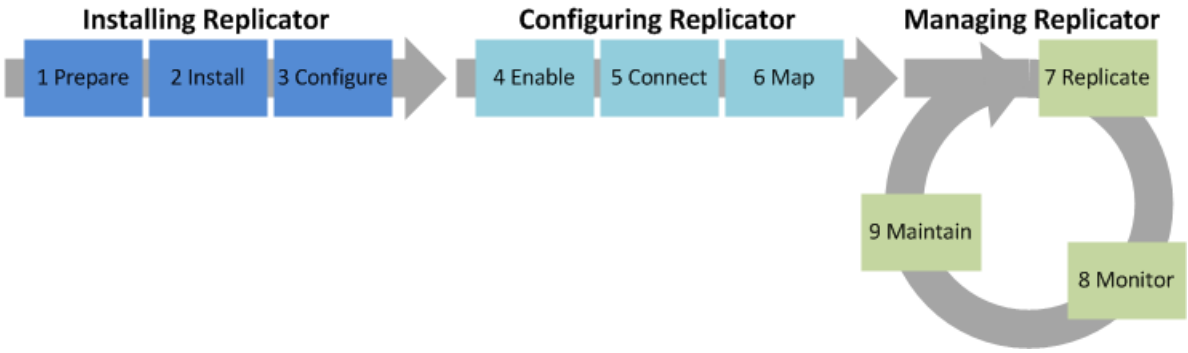
1

Introduction

This guide will provide an overview and list of all the available PowerShell commands for the configuration and management of Metalogix Replicator.

Replicator System Management Lifecycle

The Metalogix Replicator System Management Lifecycle describes our recommendation for deploying Replicator and then performing ongoing administration. The framework consists of several brief phases, which are explained in the following diagram:



This document describes the Metalogix Replicator command-line interface using the phases of the framework.

Replicator Administration

Replicator can be administered the Replicator Management Shell, which runs PowerShell cmdlets. This tool is used to:

- Complete many common administrative tasks that are available in the Replicator user interface.
- Script repetitive tasks.
- Perform long running tasks that may time out when triggered from within a web browser.
- Complete commands that are not available in the SharePoint interface.

Replicator Management Shell

The Replicator Management Shell provides Powershell commands for managing Replicator. To start the management shell, Start >Metalogix > Replicator > Replicator Management Shell. (Note, if you are running anything earlier than Windows Server 2012, you access the Replicator Management Shell by doing the following: Start> All Programs> Metalogix> Replicator> Replicator Management Shell).

The following code example shows how to enable a web application from the Replicator Management Shell:

```
Set-ReplicatorWebAppConfig -URL http://corporateoffice -Enabled 1
```

Refer to Replication Administration for the cmdlet syntax of supported administration functions.

NOTE: Specific argument tags such as -URL are not required in the management shell, but you must specify all required arguments in the order specified in this document.

Parameters with Quotes or Spaces

If you want to specify a parameter that has spaces you have to use quotes. If you want to specify a parameter that has quotes (with or without spaces) you have to escape the parameter quotes with \? and put the entire string in quotes.

Using the Management Shell in UAC-enabled Environments

If UAC is enabled on the server where Replicator is installed, then the Replicator Management Shell will not have sufficient permission to run most commands. Instead, open Windows PowerShell Modules from Administrative Tools to run these commands as an elevated user.

Manually Loading the Replicator Powershell Commands

To use the Replicator Powershell commands in an existing Powershell windows or scripts, you can use the following commands:

```
PS > C:\Windows\Microsoft.NET\Framework64\v2.0.50727\InstallUtil.exe 'C:
\Program Files\Metalogix\Replicator\WSS40\Syntergy.Replicator.Commands.dll'
PS > add-pssnapin SyntergyPsSnapin
```

Repadm Commands

It is no longer standard practice to use repadm commands to administer Replicator. We have now moved forward to using PowerShell as our main administration tool. For user that are still running their environments using repadm and have yet to switch over to PowerShell, please see the appendix at the end of this document for Repadm to PowerShell equivalents.

2

Installing Metalogix Replicator

The following sections will provide an overview of the steps required to install Metalogix Replicator, using PowerShell commands.

Prepare SharePoint Farms

There are no command-line tools for this phase of the lifecycle.

Install Metalogix Replicator

This section provides an overview on running the Metalogix Replicator for SharePoint.exe

Metalogix Replicator for SharePoint.exe

Purpose	Run the Replicator installer, with no user interface requirements.
Requirements	To run the installer without the user interface on Windows Server 2003, you must first install Windows Installer 4.5. This is available from http://download.microsoft.com.
Edition	All
Comment	The exe calls a standard Windows installer, which behaves as all standard Windows installers do. For more information on standard Windows installers, see: http://support.microsoft.com/kb/227091 or http://msdn.microsoft.com/en-us/library/aa367988.aspx

/v	Passes the subsequent parameters to the Windows Installer running beneath the executable. The subsequent parameters must be passed in quotes immediately after /v. For example /v"/log C:\install.log".
/help	Help and quick reference option. Displays the correct usage of the setup command including a list of all switches and behavior. The description of usage can be displayed in the user interface. Incorrect use of any option invokes this help option. **NOTE:** The equivalent Windows Installer Command-line Option is /?.

/quiet — Quiet display option. The installer runs an installation without displaying a user interface. No prompts, messages, or dialog boxes are displayed to the user. The user cannot cancel the installation.

NOTE: The equivalent Windows Installer Command-line Option is /qn.

/uninstall — Uninstall product option. Uninstalls a product.

NOTE: The equivalent Windows Installer Command-line Option is /x.

/log — Log option. Writes logging information into a log file at the specified existing path. The path to the log file location must already exist. The installer does not create the directory structure for the logfile.

The following information is entered into the log:

- Status messages
- Nonfatal warnings
- All error messages
- Start-up of actions
- Action-specific records
- User requests
- Initial UI parameters
- Out-of-memory or fatal exit information
- Out-of-disk-space messages
- Terminal properties

NOTE: The equivalent Windows Installer Command-line Option is /L*.

/q — No user interface is displayed

/qb — Basic user interface is displayed

/qr — Reduced user interface is displayed. A modal dialog box is displayed at the end of the installation

/qf — Full user interface is displayed. A modal dialog box is displayed at the end of the installation

/qn+ — No user interface is displayed. However, a modal dialog box is displayed at the end of the installation

/qb+	Basic user interface is displayed. A modal dialog box is displayed at the end of the installation. If you cancel the installation, a modal dialog box is not displayed
/qb-	Basic user interface with no modal dialog boxes

Examples

Silent install, using the recommended /qn+ argument that notifies the user when installation completes:

```
"Metalogix Replicator for SharePoint x64.exe" /v"/log C:\install.log /qn+"
```

Silent install, with no user interaction or notifications:

```
"Metalogix Replicator for SharePoint x64.exe" /v"/log C:\install.log /qb-"
```

Configure Metalogix Replicator

This section provides an overview on running the configwizard.exe

ConfigWizard.exe

Purpose	Run Replicator Configuration Wizard in silent mode.	
Edition	All	
Comment		
	CentralAdminAccount	Required: SharePoint Central Administration application pool identity account
	CentralAdminPassword	Required: SharePoint Central Administration application pool identity password
	LicenseKey	Required: Metalogix Replicator license key
	ActivationKeyFile	The activation key file generated for the license key at https://support.quest.com/offline-activation If you do not specify this argument, then the Configuration Wizard will perform an online activation using the license key.
	NoStartServices	Do not automatically start Replicator Service
	ResetPassword	Configure Replicator Services only. Do not deploy Replicator solutions.
	Sharename	Replicator Data Folders shared folder name
	SharedNetworkFolder	Replicator Data Folders shared network folder
	VdirConfigAccount	Replicator Data Folders virtual directory account
	VdirConfigPassword	Replicator Data Folders virtual directory password

FileSystemPath	Replicator Data Folders shared folder location
Uninstall	Run in uninstall mode

Offline Activation

If you are scripting the installation process and do not want to perform an online activation, then before running the Configuration Wizard, you must:

1. Generate activation data for your license key by running the Register-ReplicatorLicenseOffline PowerShell command.
2. Submit the activation data to https://support.quest.com/offline-activation and download the activation key file.
3. Pass the activation key file into the Configuration Wizard using the -ActivationKeyFile argument.

Examples

Run the Configuration Wizard using default settings with online activation.

```
ConfigWizard.exe -CentralAdminAccount corporate\spadmin
-CentralAdminPassword p4ssw0rd -LicenseKey 11111-22222-33333-44444-55555
```

Run the Configuration Wizard using default settings with offline activation.

```
ConfigWizard.exe -CentralAdminAccount corporate\spadmin
-CentralAdminPassword p4ssw0rd -LicenseKey 11111-22222-33333-44444-55555
-ActivationKeyFile C:\LicenseActivationResponse.dat
```

Arguments

```
[-CentralAdminAccount <domainname\username>]
[-CentralAdminPassword <password>]
[-LicenseKey <license key string>
[-Login <domainname\username>
[-NoStartServices]
[-Password <password>]
[-ResetPassword]
[-Sharename <shared folder name>]
[-SharedNetworkFolder <shared folder name>]
[-VdirConfigAccount <domainname\accountname>]
[-VdirConfigPassword <password>
[-FileSystemPath <file system path>]
[-Uninstall]
```

3

Configuring Metalogix Replicator

This section provides an overview of the all commands used for the configuration of Metalogix Replicator.

Enable Web Applications for Replication

The following are commands used in the process of enabling web applications for Replication.

Add User Mappings

AddReplicatorUserMappings

PowerShell	Add-ReplicatorUserMappings	
Purpose	Add a list of user names to be remapped by Replicator.	
Edition	Standard, Enterprise and Runtime only	
Comment	Sets the path to the list of user names and the names to map them to.	
	URL	Enter the URL to the Web Application.
	LoadPath	The full path for the XML file containing the old and new user names.
	StripDomain	Strip the domain from user when importing.

XML File Example

```
<UserRemapping>
  <User OldUser="OldUserLoginName"
     NewUser="NewUserLoginName"/>
</UserRemapping>
```

PowerShell Code Example

```
Add-ReplicatorUserMappings -URL "http://localwebapplication" -LoadPath "C:
\pathtofile.xml"
```

Returns

None

Arguments

```
[-URL] <String>
[-LoadPath] <String>
[-StripDomain [<SwitchParameter>]]
```

Configure Web Application

Set-ReplicatorWebAppConfig

PowerShell	Set-ReplicatorWebAppConfig	
Purpose	Configures the specified Web Application	
Edition	All	
Comment	This command enables Replicator on the Source Web Application.	
	URL	Enter a URL to a Web Application.
	Reset	Clears ALL Replicator Settings for this Web Application.
	Enabled	Enables or disables Replicator on the Web Application.
	CaptureEvents	Enables or disables capturing events across the entire Web Application.
	EnableProcessing	Enables or disables processing of Replication queues.
	CachePath	Set the cache location.
	ImportPath	Set the location of the Import Replicator Data Folder, where inbound packages are stored.
	ExportPath	Set the location of the Export Replicator Data Folder, where outbound packages are stored.
	PackageRetention	Sets the amount of time that packages are retained on the monitor page and Export Replicator Data Folder.
	RemapSiteOwner	Enable remapping to Site owner.
	RemapUserTable	Enable or disable remap user table.
	RemapDomainEnabled	Enables or disables domain remapping on this Web Application.
	RemapDomainName	The name to use for Domain remapping.
	RemoteExportPath	Indicates whether the Export path is on a Target Web Application.

RemoteUserAccount	User account used to access the remote export path.
RemotePassword	Password corresponding to the remote path user account.
ConfigureWfe	Configures web front-end servers for replication. You must run this command on any web front-end servers where UAC is enabled.
ChangeExternalID	Changes the external ID for this web application. This ID is only used by Replicator to uniquely identify web applications in your Replication Network. Only change this ID if you are replication between two web applications with the same GUID. After running this command, you must refresh all connections to this web application.

PowerShell Code Example

```
Set-ReplicatorWebAppConfig -URL "http://corporateoffice" -Enabled 1
```

Returns

None

Arguments

```
[-URL] <String>
[-Reset [<SwitchParameter>]]
[-Enabled [<SwitchParameter>]]
[-CachePath <String>]
[-ImportPath <String>]
[-ExportPath <String>]
[-PackageRetention <String>]
[-RemapDomainEnabled [<SwitchParameter>]]
[-DomainName <String>]
[-ConfigureWfe [<SwitchParameter>]]
[-RemoteExportPath [<SwitchParameter>]]
[-RemoteUserAccount <String>]
[-RemotePassword <String>]
[-RemapUserTable [<SwitchParameter>]]
[-CaptureEvents [<SwitchParameter>]]
[-EnableProcessing [<SwitchParameter>]]
[-PPEnabled [<SwitchParameter>]]
[-ChangeExternalID [<SwitchParameter>]]
```

Remove Replicator User Mappings

Remove-ReplicatorUserMappings

PowerShell	Remove-ReplicatorUserMappings
Purpose	Delete user mappings by Web Application or individual user.

Edition	Standard, Enterprise and Runtime only	
Comment	Removes all user mappings from the specified Web Application.	
	URL	Enter the URL to a Web Application.
	DeleteAll	Deletes all user mappings associated with the specified Web Application.
	DeleteUser	Deletes a single user mapping.

PowerShell Code Example

```
Remove-ReplicatorUserMappings -URL "http://localwebapplication" -DeleteAll
```

Returns

None

Arguments

```
[-URL] <String>
[-DeleteAll [<SwitchParameter>]]
[-DeleteUser <String>]
```

Write Default Replicator Data Folder

Write-ReplicatorDefaultDataFolder

PowerShell	Write-ReplicatorDefaultDataFolder
Purpose	Lists the UNC path used for the replicator data folder by default.
Edition	All
Comment	None

PowerShell Code Example

```
Write-ReplicatorDefaultDataFolder
```

Returns

None

Arguments

None

Write Replicator Data Folders

Write-ReplicatorDataFolders

PowerShell	Write-ReplicatorDataFolders	
Purpose	Lists the Replicator Data Folders of all URLs or of a single URL if specified.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application or a Site Collection.

PowerShell Code Example

```
Write-ReplicatorDataFolders -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
```

Update Default Replicator Data Folder

Update-ReplicatorDefaultDataFolder

PowerShell	Update-ReplicatorDefaultDataFolder	
Purpose	Updates the UNC path used for the replicator data folder by default.	
Edition	All	
Comment		
	DataFolder	Enter the UNC path to the folder to use for the default replicator data folder.

PowerShell Code Example

```
Update-ReplicatorDefaultDataFolder -DataFolder "\\UNC\Data"
```

Returns

None

Arguments

```
[-DataFolder] <String>
```

Connect Web Applications

The following are commands used in the process of connecting web applications for Replication.

Add Replication Group

Add-ReplicatorGroup

PowerShell	Add-ReplicatorGroup	
Purpose	Add new Replication Group to the URL.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter the URL to the Web Site.
	Group	Enter a name for the Replication Group being created.
	AutoAddNewConnections - -	Specify if new Replication Connections will automatically become members of this Replication Group.
	DefaultGroup	Enable or disable this group as the default Replication Group for the new Map Family.
	AllowCustomBinding	Allow or disallow Custom Binding for Connections in this Replication Group.
	ReplicationMode	Enter Direct mode to replicate packages only using defined connections or Adaptive mode to have Replicator find the best path between pairs of connected web applications.

PowerShell Code Example

```
Add-ReplicatorGroup -URL "http://localwebapplication" -Group "Test
Replication Group Name" -DefaultGroup
```

Returns

None

Arguments

```
[-URL] <String>
[-Group] <String>
[-DefaultGroup [<SwitchParameter>]]
[-AutoAddNewConnections [<SwitchParameter>]]
[-AllowCustomBinding [<SwitchParameter>]]
[-ReplicationMode <String>]
```

Add Replicator Group Connection

Add-ReplicatorGroupConnection

PowerShell	Add-ReplicatorGroupConnection	
Purpose	Add a new Connection to the specified Replication Group.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter the URL to the Web Application.
	Group	Enter a name for the Replication Group to add the Connection to.
	Connection	Enter a name for the Connection to add to the Replication Group.
	InputConnection	Enter a Replication Connection to add to the Replication Group

PowerShell Code Example

```
Add-ReplicatorGroupConnection -URL "http://localwebapplication" -Group
"Test Replication Group" -Connection "Test Connection Name"
```

Returns

None

Arguments

```
[[-URL] <String>]
[-Group] <String>
[[-Connection] <String>]
[-InputConnection <Server>]
```

Remove Replicator Group

Remove-ReplicatorGroup

PowerShell	Remove-ReplicatorGroup	
Purpose	Delete the specified Replication Group.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application.
	Group	Enter a name for the Replication Group to delete.
	InputGroup	Enter a name for the Replication Group to delete.

PowerShell Code Example

```
Remove-ReplicatorGroup -URL "http://localwebapplication" -Group "Test
Replication Group Name"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-Group] <String>]
[-InputGroup <ReplicationGroup>]
```

Remove Replicator Group Connection

Remove-ReplicatorGroupConnection

PowerShell	Remove-ReplicatorGroupConnection	
Purpose	Delete an existing Connection from the specified Replication Group.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application.
	Group	Enter a name for the Replication Group to remove the Connection from.

Connection	Enter a name for the Connection to remove from the Replication Group.
InputConnection	Enter a name for the Connection to remove from the Replication Group.

PowerShell Code Example

```
Remove-ReplicatorGroupConnection -URL "http://localwebapplication" -Group
"Test Replication Group" -Connection "Test Connection Name"
```

Returns

None

Arguments

```
[[URL] <String>]
[-Group] <String>
[[-Connection] <String>]
[-InputConnection <Server>]
```

Set Replicator RDC Cache

Set-ReplicatorRDCCache

PowerShell	Set-ReplicatorRDCCache	
Purpose	Configures the RDC Cache.	
Edition	All	
Comment	This command configures the RDC Cache for the specified Web Application.	
	URL	Enter a URL to a Web Application.
	ConsumePath	Path to directory tree from which to build the cache.
	ConsumeURL	URL to Web Site from which to build the cache.
	ConsumeURLListName	Name of the Web Site list from which to build the cache.
	ClearExisting	Clears the existing cache prior.
	Force	Run even if cache is locked.

PowerShell Code Example

```
Set-ReplicatorRDCCache -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-ConsumePath <String>]
[-ConsumeURL <String>]
[-ConsumeURLListName <String>]
[-ClearExisting [<SwitchParameter>]]
[-Force [<SwitchParameter>]]
```

Set Replicator RDC Dynamic Cache

Set-ReplicatorRDCDynamicCache

PowerShell	Set-ReplicatorRDCDynamicCache	
Purpose	Configures RDC Dynamic Cache.	
Edition	All	
Comment	This command configures the RDC Dynamic Cache for the specified Web Application.	
	URL	Enter a URL to a Web Application.
	ClearExisting	Clears the existing cache prior.
	ClearDate	Only clears that is older than specified date.

PowerShell Code Example

```
Set-ReplicatorRDCDynamicCache -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-ClearExisting [<SwitchParameter>]]
[-ClearDate <String>]
```

Update Replicator Connection

Update-ReplicatorConnection

PowerShell	Update-ReplicatorConnection
Purpose	Updates configuration for Replication Connections.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to the Web Application.
ConnectionName	The name of the connection to update.
TargetDomain	The domain for the target user account.
TargetUserName	The currently configured user name of the application pool account on the Target Web Application.
TargetPassword	The password for the target user account.
PauseOutboundPackaging	Pauses/Resumes inbound package processing on this connection.
PauseInboundProcessing	Pauses/Resumes inbound package processing on this connection
EnableEventCapture	Enables/Disables event capture on this connection.
ConflictHandling	Specifies what action is to be taken on inbound Events when a conflict is detected.
EmailWebSiteAdmin	Sends email notifications to the Web Site Administrator.
EmailItemUpdater	Sends email notifications to the Item Updater.
EmailItemOwner	Sends email notifications to the Item Owner.
EmailSpecificAddress	Send an email notification to the specified address. Type "" for unchecking this option.
MaximumRetryCount	Specifies the maximum number of times Replicator should retry processing and sending a package.
EnablePauseConnection	Specifies if processing should be periodically paused during the retry process.
RetryCountBeforePausing	Specifies the number of times Replicator should retry a package before pausing.
PauseDuration	Specifies the pause duration in minutes.
MonitorUpdateLevel	Specifies a monitor update level that is appropriate for your network.

PowerShell Code Example

```
Update-ReplicatorConnection -URL "http://corporateoffice" -ConnectionName
"CorporateToLondonConnection" -TargetDomain "londonoffice" -TargetUserName
"spadmin" -TargetPassword "pa55w0rd"
```

Returns

None

Arguments

```
[-URL] <String>
[-ConnectionName] <String>
[-TargetDomain <String>]
[-TargetUserName <String>]
[-TargetPassword <String>]
[-PauseOutboundPackaging [<SwitchParameter>]]
[-PauseInboundProcessing [<SwitchParameter>]]
[-EnableEventCapture [<SwitchParameter>]]
[-ConflictHandling <String>]
[-EmailWebSiteAdmin <Boolean>]
[-EmailItemUpdater <Boolean>]
[-EmailItemOwner <Boolean>]
[-EmailSpecificAddress <String>]
[-MaximunRetryCount <String|Int32>]
[-EnablePauseConnection <Boolean>]
[-RetryCountBeforePausing <Int32>]
[-PauseDuration <Int32>]
[-MonitorUpdateLevel <String>]
```

Update Replicator Connections

Update-ReplicatorConnections

PowerShell	Update-ReplicatorConnections	
Purpose	Refreshes one or all connections for a web application. This command can quickly update all connections after its target web applications have been upgraded.	
Edition	All	
Comment		
	URL	Enter the URL to the Web Application.
	ConnectionName	Enter a name of a specific connection to refresh.

PowerShell Code Example

```
Update-ReplicatorConnections -URL http://corporateoffice -ConnectionName
"Corporate to London Connection"
```

Returns

```
Connection <ConnectionName> has been refreshed
```

Arguments

```
[-URL] <String>
[-ConnectionName <String>]
```

Update Replicator Group

Update-ReplicatorGroup

PowerShell	Update-ReplicatorGroup	
Purpose	Update an existing Replication Group with the specified settings.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application.
	Group	Enter a name for the Replication Group to update.
	AutoAddNewConnections	Specify if new Replication Connections will automatically become members of this Replication Group.
	DefaultGroup	Enable or disable default Replication Group.
	AllowCustomBinding	Allow or disallow Custom Binding.
	InputGroup	Enter a name for the Replication Group to update.

PowerShell Code Example

```
Update-ReplicatorGroup -URL "http://localwebapplication" -Group "Test
Replication Group Name" -DefaultGroup -AllowCustomBinding
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-Group] <String>]
[-DefaultGroup [<SwitchParameter>]]
[-AutoAddNewConnections [<SwitchParameter>]]
[-AllowCustomBinding [<SwitchParameter>]]
[-InputGroup <ReplicationGroup>]
```

Map SharePoint Content for Replication

The following are commands used in the process of mapping SharePoint content for Replication.

Add Replicator List Mappings

Add-ReplicatorListMappings

PowerShell	Add-ReplicatorListMappings	
Purpose	Configures Replicator to replicate changes from a source list to a target list with a different name.	
Edition	Enterprise and Standard only	
Comment	Used on the outbound replication server only.	
	URL	Enter a URL to a Web Application, Site Collection or a Web Site.
	MapFamily	Enter a name for the Map Family to find the map to add list mappings to.
	Type	Specify whether the URL is to a Web Application, Site Collection or a Web Site. If this parameter is not specified, the map family will be interpreted as a Web Site.
	LoadPath	The full path and name of the XML file describing the list mapping.

XML File Example

```
<ListRemappings>
      <ListRemapping SourceList="Shared Documents" DestList="Corporate
Docs"/>
</ListRemappings>
```

PowerShell Code Example

```
Add-ReplicatorListMappings -URL "http://localwebapplication" -LoadPath "C:
\listmappings.xml"
```

Returns

None

Arguments

```
[-URL] <String>
[-LoadPath] <String>
[-Type <String>]
-MapFamily <String>
```

Add Replicator Map Family

Add-ReplicatorMapFamily

PowerShell	Add-ReplicatorMapFamily	
Purpose	Add a new Map Family with the specified settings.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter the URL to the Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter a name for the Map Family.
	Type	Specify whether the URL is to a Web Application, Site Collection, Web Site, List, or Folder.
	Group	Enter a name for the Group being set. If you do not specify a Group, then the default group will be used. If there is no default group, then you must specify the group with this command. (optional)
	Enabled	Enable or disable the Map Family when created. (optional)
	MaxEventCount	Specify the maximum number of events that will be included in a replication package.

PowerShell Code Example

```
Add-ReplicatorMapFamily -URL "http://localwebapplication" -Type
"WebApplication" -MapFamily "Test Map" -Group "Test Group Name"
```

Returns

```
Map Family added.
```

Arguments

```
[-URL] <String>
[-MapFamily] <String>
[-Enabled [<SwitchParameter>]]
[-Group <String>]
[-MaxEventCount <Int32>]
[-Type] <String>
```

Remove Replicator List Mappings

Remove-ReplicatorListMappings

PowerShell	Remove-ReplicatorListMappings	
Purpose	Delete list Mappings	
Edition	Enterprise and Standard only	
Comment		
	URL	Enter a URL to a Web Application, Site Collection or a Web Site.
	Type	Specify whether the URL is to a Web Application, Site Collection or a Web Site. If this parameter is missing, the URL will be interpreted as a Web Site.
	MapFamily	Enter a name for the Map Family

PowerShell Code Example

```
Remove-ReplicatorListMappings -URL "http://localwebapplication" -MapFamily
"Map Family Name"
```

Returns

None

Arguments

```
[-URL] <String>
[-MapFamily] <String>
[-Type <String>]
```

Remove Replicator Map Family

Remove-ReplicatorMapFamily

PowerShell	Remove-ReplicatorMapFamily	
Purpose	Delete the specified Map Family.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter the name for the Map Family.
	InputMapFamily	Enter the name for the Map Family.

PowerShell Code Example

```
Remove-ReplicatorMapFamily -URL "http://localwebapplication" -MapFamily
"Portal Map Family"
```

Returns

None

Arguments

```
[[-URL] <String>]
[-MapFamily <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Disable Replicator Alert Replication

Disable-ReplicatorAlertReplication

PowerShell	Disable-ReplicatorAlertReplication	
Purpose	Disable replication for alert events, such as a user configuring alerts on a list.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a web application, site collection or a web site. Alerts for this URL and any child sites will no longer be replicated.
	Type	Specify whether the URL is to a web application, site collection or a web site. The default type is web site.

PowerShell Code Example

```
Disable-ReplicatorAlertReplication -URL "http://localwebapplication" -Type
SiteCollection
```

Returns

None

Arguments

```
[-URL] <String>
[-Type <String>]
```

Disable Replicator Map

Disable-ReplicatorMap

PowerShell	Disable-ReplicatorMap	
Purpose	Disables the Replication Map associated with a specified URL and Map Family, if specified.	
Edition	Standard, Enterprise, and Runtime only	
Comment	Disables the Replication Maps associated with a specified URL and Map Family, if specified.	
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	Type	Specify whether the URL is to a Web Application, Site Collection, Web Site, List, or Folder.
	IncludeChildren	Include child Web Sites.
	MapFamily	Enter a name for a Map Family.
	InputReplicatorMap	Enter a name for a Map Family.

PowerShell Code Example

```
Disable-ReplicatorMap -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[-Type <String>]
[-IncludeChildren [<SwitchParameter>]]
[[-MapFamily] <String>]
[-InputReplicationMap <ReplicationMap>]
```

Disable Replicator Map Family

Disable-ReplicatorMapFamily

PowerShell	Disable-ReplicatorMapFamily	
Purpose	Disables the specified Map Family.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter a name for the Map Family to disable.
	InputMapFamily	Enter a name for the Map Family to disable.

PowerShell Code Example

```
Disable-ReplicatorMapFamily -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Suspend Replicator Map Family

Suspend-ReplicatorMapFamily

PowerShell	Suspend-ReplicatorMapFamily
Purpose	Suspends Replication for the specified Map Family.
Edition	Standard, Enterprise, and Runtime only

Comment		
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter a name for the Map Family to disable.
	InputMapFamily	Enter a name for the Map Family to disable.

PowerShell Code Example

```
Suspend-ReplicatorMapFamily "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Resume Replicator Map Family

Resume-ReplicatorMapFamily

PowerShell	Resume-ReplicatorMapFamily	
Purpose	Resumes Replication for the specified Map Family.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter a name for the Map Family to disable.
	InputMapFamily	Enter a name for the Map Family to disable.

PowerShell Code Example

```
Resume-ReplicatorMapFamily "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Enable Replicator Alert Replication

Enable-ReplicatorAlertReplication

PowerShell	Enable-ReplicatorAlertReplication	
Purpose	Enable replication for alert events, such as a user configuring alerts on a list.	
Edition	All	
Comment		
	URL	Enter a URL to a web application, site collection or a web site. Alerts for this URL and any child sites will be replicated.
	Type	Specify whether the URL is to a web application, site collection or a web site. The default type is web site.

PowerShell Code Example

```
Enable-ReplicatorAlertReplication -URL "http://localwebapplication" -Type
"WebApplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-Type <String>]
```

Enable Replicator Map

Enable-ReplicatorMap

PowerShell	Enable-ReplicatorMap
Purpose	Enables all the Replication Maps associated with the specified URL and Map Family, if specified.
Edition	Standard, Enterprise, and Runtime only

Comment

URL	Enter a URL to a Web Application, a Site Collection, Web Site, List, or Folder.
Type	Specify whether the included URL is to a Web Application, Site Collection, Web Site, List, or Folder.
IncludeChildren	Enable the Map Families for the child Web Sites.
MapFamily	Enter a name for a Map Family.
InputReplicationMap	Enter a name for a Map Family.

PowerShell Code Example

```
Enable-ReplicatorMap -URL "http://localwebapplication" -IncludeChildren
```

Returns

None

Arguments

```
[[-URL] <String>]
[-Type <String>]
[-IncludeChildren [<SwitchParameter>]]
[[-MapFamily] <String>]
[-InputReplicationMap <ReplicationMap>]
```

Enable Replicator Map Family

Enable-ReplicatorMapFamily

PowerShell	Enable-ReplicatorMapFamily
Purpose	Enables the specified Map Family.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
MapFamily	Enter a name for the Map Family to enable.
InputMapFamily	Enter a name for the Map Family to enable.

PowerShell Code Example

```
Enable-ReplicatorMapFamily -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Write Replicator Map

Write-ReplicatorMap

PowerShell	Write-ReplicatorMap	
Purpose	Display the information for a specified Replication Map at the specified Web site.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
	MapFamily	Enter a name for the Map Family.
	Type	Specify whether the URL is to a Web Application, Site Collection, Web Site, or Folder.

PowerShell Code Example

```
Write-ReplicatorMap -URL "http://corporateoffice" -MapFamily "Portal Map
Family" -Type WebApplication
```

Returns

None

Arguments

```
[-URL] <String>
[-MapFamily] <String>
-Type <String>
```

Write Replicator Map Families

Write-ReplicatorMapFamilies

PowerShell	Write-ReplicatorMapFamilies	
Purpose	Display a list of all the Map Families on the specified Web Application.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application.
	IncludeChildren	Include child Web Sites.

PowerShell Code Example

```
Write-ReplicatorMapFamilies -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-IncludeChildren [<SwitchParameter>]]
```

Suspend Replicator Map Families

Suspend-ReplicatorMapFamilies

PowerShell	Suspend-ReplicatorMapFamilies
Purpose	Suspends replication for all Replication Map Families on a web application.
Edition	Standard, Enterprise, and Runtime only
Comment	

	URL	Enter a URL to a Web Application.

PowerShell Code Example

```
Suspend-ReplicatorMapFamilies "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
```

Write Replicator Paused Map Families

Write-ReplicatorPausedMapFamilies

PowerShell	Write-ReplicatorPausedMapFamilies	
Purpose	Display a list of all the suspended Map Families on the specified Web Application.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Application.

PowerShell Code Example

```
Write-ReplicatorPausedMapFamilies "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
```

Resume Replicator Map Families

Resume-ReplicatorMapFamilies

PowerShell	Resume-ReplicatorMapFamilies
Purpose	Resumes replication for all Replication Map Families on a web application.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
MapFamily	Enter a name for the Map Family to disable.
InputMapFamily	Enter a name for the Map Family to disable.

PowerShell Code Example

```
Resume-ReplicatorMapFamilies "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Set Replicator Map Family

Set-ReplicatorMapFamily

PowerShell	Set-ReplicatorMapFamily
Purpose	Resets all the Map Families associated with the URL.
Edition	Standard, Enterprise, and Runtime only
Comment	

	URL	Enter a URL to a Web Application, Site Collection, or a Web Site.
	Map Family	Enter the name of the Map Family
	InputMapFamily	Enter the name of the Map Family

PowerShell Code Example

```
Set-ReplicatorMapFamily -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Write Replicator Maps

Write-ReplicatorMaps

PowerShell	Write-ReplicatorMaps	
Purpose	Display a list of all the Replication Maps at the specified Web site.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter a URL to a Web Site.
	IncludeChildren	Include child Web Sites.

PowerShell Code Example

```
Write-ReplicatorMaps -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-IncludeChildren [<SwitchParameter>]]
```

Set Replicator Map Inheritance

Set-ReplicatorMapInheritance

PowerShell	Set-ReplicatorMapInheritance	
Purpose	Resets all R eplication Maps associated with a Full Replication starting with the specified URL and continuing to all the system created children.	
Edition	All	
Comment	This command enables inheritance for the specified URL and all its children.	
	URL	Enter a URL to a Web Application, Site Collection or a Web Site.
	MapFamily	Enter the name for the Map Family.
	IncludeChildren	Include child Replication Maps.
	Type	Specify whether the included URL is to a Web Application, Site Collection or a Web Site

PowerShell Code Example

```
Set-ReplicatorMapInheritance -URL "http://localwebapplication" -
IncludeChildren
```

Returns

None

Arguments

```
[-URL] <String>
[[-IncludeChildren] [<SwitchParameter>]]
-Type <String>
[[-MapFamily] <String>]
```

Update Replicator Map

Update-ReplicatorMap

PowerShell	Update-ReplicatorMap
Purpose	Update an existing Replication Map with the specified settings.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
MapFamily	Enter a name for the Map Family.
Type	Specify whether the URL is to a Web Application, Site Collection, Web Site, List, or Folder.
Enabled	Enable or disable the Replication Map when created.
Inherit	Enable or disable inheritance on the Replication Map when created.
ChildInherit	Enable or disable inheriting child maps.
BreakInheritance	Explicitly breaks inheritance for use with event modifications.
AddEventsByName	Specify the event names that you want to capture with this map family. The event names are listed under Supported Events for the Queue Map command in this guide. Separate multiple event names with a semi-colon. For example, "List Add;List Item Add".
RemoveEventsByName	Specify the event names that you do not want to capture with this map family. The event names are listed under Supported Events for the Queue Map command in this guide. Separate multiple event names with a semi-colon. For example, "List Add;List Item Add".
AddEventsByGroupName	Specify the event group names that you want to capture with this map family. All events in that group will be captured. Valid values are Web Application, Site Collection, Web Site, List, List Item, Workflow, and Look and Feel. Separate multiple group names with a semi-colon. For example "List;List Item".
RemoveEventsByGroupName	Specify the event group names that you do want to capture with this map family. No events in that group will

	be captured. Valid values are Web Application, Site Collection, Web Site, List, List Item, Workflow, and Look and Feel. Separate multiple group names with a semi-colon. For example "List;List Item".
AddAllEvents	Adds all events in the current event pool to the selected events list for this map.
RemoveAllEvents	Removes all events in the selected events list for this map.
InputReplicationMap	Enter the map family name.

PowerShell Code Example

```
Update-ReplicatorMap -URL "http://corporateoffice" -MapFamily "Portal Map
Family" -Type WebSite -RemoveEventsByGroupName "Workflow:Look and Feel"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-Type <String>]
[-Enabled [<SwitchParameter>]]
[-Inherit [<SwitchParameter>]]
[-ChildInherit [<SwitchParameter>]]
[-AddEventsByName <String>]
[-RemoveEventsByName <String>]
[-AddEventsByGroupName <String>]
[-RemoveEventsByGroupName <String>]
[-AddAllEvents [<SwitchParameter>]]
[-RemoveAllEvents [<SwitchParameter>]]
[-BreakInheritance [<SwitchParameter>]]
[-InputReplicationMap <ReplicationMap>]
```

Update Replicator Map Connection

Update-ReplicatorMapConnection

PowerShell	Update-ReplicatorMapConnection
Purpose	Update an existing Replication Connection with the specified settings.
Edition	Standard, Enterprise, and Runtime only

Comment

URL	Enter a URL to a Web Site.
MapFamily	Enter the name for the Replication Map.
Connection	Only queue for the specified Connection.
Type	Specify whether the URL is to a Web Application, Site Collection or a Web Site.
Enabled	Enable or disable the Replication Map when created.
TargetRelativeURL	A Connection relative URL to the Target Web Application to bind to.

PowerShell Code Example

```
Update-ReplicatorMapConnection -URL "http://localwebapplication" -
Connection "Connection Name" -MapFamily "Map Family Name" -Enabled
```

Returns

None

Arguments

```
[-URL] <String>
[-MapFamily] <String>
-Type <String>
[-Connection] <String>
[-Enabled [<SwitchParameter>]]
[-TargetRelativeURL <String>]
```

Update Replicator Map Family

Update-ReplicatorMapFamily

PowerShell	Update-ReplicatorMapFamily
Purpose	Update an existing Map Family with the specified settings.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.

MapFamily	Enter a name for the Map Family.
Enabled	Enables or disables the specified map family. If you do not specify this value then the status of the map family will not change. (optional)
InputMapFamily	Enter the name of the Map Family.
MaxEventCount	Specify the maximum number of events that will be included in a replication package.
Pause	Pauses or resumes the specified map family.

PowerShell Code Example

```
Update-ReplicatorMapFamily -URL http://localwebapplication -MapFamily "My Portal"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-Enabled <Boolean>]
[-InputMapFamily <ReplicationMapFamily>]
[-MaxEventCount <Int32>]
[-Pause <SwitchParameter>]
```

Update Replicator Map Lists

Update-ReplicatorMapLists

PowerShell	Update-ReplicatorMapLists
Purpose	Enable or disable lists in an existing Replication Map.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a Web Application.
MapFamily	Enter a name for the Map Family.
Type	Specify whether the URL is for a Web Application, Site Collection or a Web Site.

Disable	Disable all lists in the Replication Map. (optional)
SetReplicateAllLists	Enable all lists in the Replication Map. (optional)

PowerShell Code Example

```
update-replicatormaplists -URL "http://localwebapplication" -Disable -
MapFamily "Map Family Name"
```

Returns

None

Arguments

```
[-URL] <String>
[-Type <String>]
-MapFamily <String>
[-Disable [<SwitchParameter>]]
[-SetReplicateAllLists [<SwitchParameter>]]
```

Get Replicator Objects Commands

The following are commands used for retrieving replicator objects.

Get Replicator Web Application

Get-ReplicatorWebApp

PowerShell	Get-ReplicatorWebApp	
Purpose	Returns Replicator Web Application objects. Replicator has to have been enabled at least once on the Web Application.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	Enter the URL of the web application that is to be returned.
	Name	Enter the name of the publish server that is to be returned. Wildcard characters can be used in the name argument.
Cmdlet Piping Consumer	Write-ReplicatorUserMappings	

Write-ReplicatorDataFolders

Write-ReplicatorCounters

Update-ReplicatorGroup

Update-ReplicatorMap

Update-ReplicatorConnections

Update-ReplicatorConnection

Start-ReplicatorMaintenance

Set-ReplicatorWebAppConfig

Remove-ReplicatorUserMappings

Save-ReplicatorCurrentCounters

Set-ReplicatorRDCCache

Set-ReplicatorRDCDynamicCache

Remove-ReplicatorGroup

Remove-ReplicatorGroupConnection

Export-ReplicatorPackages

Import-ReplicatorPackages

Reapply-ReplicatorErrorQueueItems

Clear-ReplicatorWebAppConfig

Clear-ReplicatorQueueCompletedOrError

Clear-ReplicatorQueueCompleted

Clear-ReplicatorQueueAll

Add-ReplicatorUserMappings

Add-ReplicatorGroupConnection

Add-ReplicatorGroup

Add-ReplicatorMapFamily

PowerShell Code Example

```
Get-ReplicatorWebApp -URL "http://corporateoffice"
```

Returns

Returns all Replicator web application objects for the specified web applications.

Arguments

```
[[-URL] <String>]
[-Name <String>]
```

Piping Example

```
Get-ReplicatorWebApp | Add-ReplicatorGroup -Group "MyCustomGroup" -
ReplicationMode "Direct"
```

Get Replicator Group

Get-ReplicatorGroup

PowerShell	Get-ReplicatorGroup	
Purpose	Returns Replicator Group objects.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	The URL of the server where the groups are configured.
	Name	The name you wish to match the returned groups against. Wildcard characters can be used in the name argument.
Cmdlet Piping Consumer	Update-ReplicatorGroup	
	Remove-ReplicatorGroup	

PowerShell Code Example

```
Get-ReplicatorGroup -URL "http://corporateoffice"
```

Returns

Returns all Replicator group objects for the specified groups.

Arguments

```
[[-URL] <String>]
[-Name <String>]
```

Piping Example

```
Get-ReplicatorGroup -Name *Custom* | Update-ReplicatorGroup -
AutoAddNewConnections:$true -AllowCustomBinding:$False
```

Get Replicator Map Family

Get-ReplicatorMapFamily

PowerShell	Get-ReplicatorMapFamily	
Purpose	Returns Replicator Map Family objects.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	The URL of the server that has the map family.
	Name	The name you wish to match the returned map families against. Wildcard characters can be used in the name argument.
Cmdlet Piping Consumer	Disable-ReplicatorMapFamily	
	Enable-ReplicatorMapFamily	
	Remove-ReplicatorMapFamily	
	Set-ReplicatorMapFamily	
	Start-ReplicatorProcessing	
	Update-ReplicatorMapFamily	

PowerShell Code Example

```
Get-ReplicatorMapFamily -Name "Map*"
```

Returns

Returns all Replicator Map Family objects for the specified map families.

Arguments

```
[[-URL] <String>]
[-Name <String>]
```

Piping Example

```
Get-ReplicatorMapFamily -URL http://corporate | Disable-ReplicatorMapFamily
```

Get Replicator Map

Get-ReplicatorMap

PowerShell	Get-ReplicatorMap	
Purpose	Returns Replicator Map objects.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	The URL of the server that has the Replicator map.
	Name	The name you wish to match the returned maps against. Wildcard characters can be used in the name argument.
Cmdlet Piping Consumer	Disable-ReplicatorMap	
	Enable-ReplicatorMap	
	Update-ReplicatorMap	

PowerShell Code Example

```
Get-ReplicatorMap -URL "http://corporateoffice"
```

Returns

Returns all Replicator Map objects for the specified maps.

Arguments

```
[[-URL] <String>]
[-Name <String>]
```

Piping Example

```
Get-ReplicatorMap -URL http://corporate | Disable-ReplicatorMap
```

Get Replicator Connection

Get-ReplicatorConnection

PowerShell	Get-ReplicatorConnection	
Purpose	Returns Replicator Connection objects.	
Edition	Standard, Enterprise, and Runtime only	
Comment		
	URL	The URL of the server that has the Replicator Connection.
	Name	The name you wish to match the returned replicator connection against. Wildcard characters can be used in the name argument.
Cmdlet Piping Consumer	Add-ReplicatorGroupConnection	
	Remove-ReplicatorGroupConnection	
	Update-ReplicatorMap	

PowerShell Code Example

```
Get-ReplicatorConnection -Name "Corporate*"
```

Returns

Returns all Replicator Connection objects for the specified connections.

Arguments

```
[[-URL] <String>]
[-Name <String>]
```

Piping Example

```
Get-ReplicatorConnection -URL http://corporate | Add-
ReplicatorGroupConnection -ReplicatorGroup CustomGroup
```

Disable Replicator

Disable-Replicator

PowerShell	Disable-Replicator	
Purpose	Disables all web applications that have been Replicator enabled.	
Edition	Standard, Enterprise, and Runtime only	
Comment	This command can only be run by the application pool account.	
	UserAccount	The user account for the application pool account.
	Password	The Password for the user accounts
	Reset	Clears all replication settings for this web application.

PowerShell Code Example

```
Disable-Replicator -UserAccount "spsadmin" -password "123456" -Reset
```

Returns

None

Arguments

```
[-UserAccount] <String>
[-Password] <String>
[[-Reset] [<SwitchParameter>]]
```

4

Managing Metalogix Replicator

This section provides an overview of the all commands used for the management of Metalogix Replicator.

Replicate SharePoint Content

The following are commands used for replicating SharePoint content.

Export Replicator Packages

Export-ReplicatorPackages

PowerShell	Export-ReplicatorPackages	
Purpose	Exports Replication Packages for Web Applications running in Offline Mode.	
Edition	Standard, Enterprise, and Runtime only	
Comment	Exports content to Packages, which can be delivered offline to the intended Web Application.	
	URL	The URL of the web application you are exporting from.
	Connection	The Replication Connection name from this web application to the target web application.
	ExportPath	The path to save the Replication Packages to. If the specified directory does not exist, then this command will create the directory.
	Force	Continues exporting subsequent packages after an error occurs.

PowerShell Code Example

```
Export-ReplicatorPackages -URL "http://corporateoffice" -Connection
"Corporate to London" -ExportPath "C:\Export"
```

Returns

None

Arguments

```
[-URL] <String>
[-Connection] <String>
[-ExportPath] <String>
[-Force [<SwitchParameter>]]
```

Import Replicator Packages

Import-ReplicatorPackages

PowerShell	Import-ReplicatorPackages	
Purpose	Imports Replication Packages for Web Applications running in Offline Mode.	
Edition	All	
Comment	Imports content from Packages that have been delivered offline to the intended Source Web Application.	
	URL	The URL for the web application you are importing and applying the Replication Packages to.
	Connection	The Connection name from this web application to the source web application.
	ImportPath	The path to import Replication Packages from. If you are importing multiple sets of exported packages, then you must import them in the order they were exported.
	Force	Continues importing subsequent packages after an error occurs.
	Adaptive	Overrides the manifest, forcing all packages to be adaptive.

PowerShell Code Example

```
Import-ReplicatorPackages -URL "http://londonoffice" -Connection "London to
Corporate" -ImportPath "C:\Import"
```

Returns

None

Arguments

```
[-URL <String>]
[-Connection <String>]
-ImportPath <String>
[-Force [<SwitchParameter>]]
[-Adaptive <Boolean>]
```

Schedule Replicator Map

Schedule-ReplicatorMap

PowerShell	Schedule-ReplicatorMap
Purpose	Schedules Replication Map and its inheriting children for replication.
Edition	All
Comment	Schedules the map at the specified URL, including the child Web Sites. When queuing large amounts of content, we recommend separating the content into batches. For example, you can queue only site structure, then permissions, and then list items. Or you can queue one Site Collection or Web Site at a time. Schedule-ReplicatorMap is not aware of any list dependencies. If you have a lookup list that is used by other lists, queue the lookup list ahead of any list that depends on it. Queuing the "Add" events will queue all available entities for that event from the source. If the entity exists on the target, then the entity will be updated, if the entity does not exist on the target, then the entity will be added.

URL	Enter a URL to a Web Application, Site Collection, Web Site, List, or Folder.
JobName	Enter a name for the schedule job.
MapFamily	Queue using the Replication Map from specified Map Family. (optional)
IncludeChildTypes	Determines which child types are to be including when queuing a map. Child types include: 'WebApplication', 'SiteCollection', 'WebSite', 'List', and 'Folder'. Only events at the root level will be queued for replication. (optional)
Type	Specify whether the URL is to a Web Application, Site Collection, Web Site, List, or Folder. If you do not specify a

	type, then the URL will be interpreted as a Web Site. (optional)
Connection	Only queue for the specified Connection. (optional)
OnlySiteStructure	Include only Site Collection and Web Site creation events. (optional)
ExcludeSiteStructure	Exclude Site Collection and Web Site creation events. (optional)
OnlyPermissions	Replicate only Web Site permissions. (optional)
OnlyMaps	Replicate only Replication Maps. (optional)
OnlyListItems	Replicate only list items and folders. (optional)
ListItemID	ID of individual list item to queue. (optional)
IgnoreConflicts	Ignore possible conflicts at the destination. (optional)
OnlyLastDays	Replicate only list items from the specified last days (excluding folders). (optional)
OnlyAlerts	Replicate only Alerts. (optional)
OnlyWFI	Replicate only Workflow associations and instances. (optional)
OnlyLookAndFeel	Replicate only Look and Feel events. (optional)
ExcludeLookAndFeel	Does not replicate Look and Feel events. (optional)
QueueVersions	Include All Versions. This will delete the list item or document specified, and then resend it with all the current versions on the source. (optional)
ClearList	Clear all items in the list. (optional)
QueueSetName	Tag all the events queued with the specified name. (optional)
Classification	Overrides the default Map Family classification.
EventSettingsXML	Specify the location of the XML with settings to use for the Queueing. This override settings that may be located on the Replication Map.
BackupMode	Uses the SharePoint site collection backup feature to create a backup of the specified site collection which is then packaged up and replicated to the target where the backup is then restored on the target, without remapping structure, domains, or users. If you do not specify a value, then backup mode is disabled. This option can only be

	used with Web Application and Site Collection map families. (optional) **Caution:** Using this option overwrites site collections on the target farm.
IncludeEvents	Replicate only the specified events. You must specify the events, by number, as a semi-colon delimited list. For the list of event numbers, see "Supported Events".
ExcludeEvents	Replicate all events specified in the map family, except for the specified events. You must specify the events, by number, as a semi-colon delimited list. For the list of event numbers, see "Supported Events".
ScheduleType	Specify the type of schedule to be used for this job.
SpecificDateTime	Specify the specific date and time to perform replication. Use with the SpecificDateTime Schedule Type.
Date	Specify the date to perform the replication. Use with Monthly Schedule Type.
Day	Specify the day of the week to perform the replication. Use with Weekly Schedule type.
Hour	Specify the hour of the day to perform the replication. Use with the Monthly, Weekly, and Daily Schedule Type.
Minute	Specify how many minutes past the hour to perform the replication. Use with the Monthly, Weekly, and Daily Schedule Types.

PowerShell Code Example

```
Schedule-ReplicatorMap -URL "http://localwebapplication" -Type
Website -IncludeChildTypes List,Folder -JobName "Test"
```

Returns

None

Arguments

```
[-Url] <String>
[[-JobName] <String>]
[-IncludeChildTypes <String>]
[-Connection <String>]
[[-MapFamily] <String>]
[-Type <String>]
```

```
[-OnlySiteStructure]
[-ExcludeSiteStructure]
[-OnlyPermissions]
[-OnlyListItems]
[-OnlyMaps]
[-OnlyAlerts]
[-OnlyLookAndFeel]
[-ExcludeLookAndFeel]
[-QueueVersions]
[-EventSettingsXML <String>]
[-ClearList]
[-IncludeEvents <String>]
[-ExcludeEvents <String>]
[-OnlyWFI]
[-ListItemID <Int32>]
[-QueueSetName <String>]
[-BackupMode [<Boolean>]]
[-Classification <String>]
[-IgnoreConflicts]
[-OnlyLastDays <Int32>]
[-ScheduleType [<String>]]
[-SpecificDateTime
[<string>]] [-Date [<Int32>]]
[-Day [<String>]]
[-Hour [<Int32>]]
[-Minute [<Int32>]]
```

Supported Events

The following table shows the event IDs that are associated with the events for the IncludeEvents and ExcludeEvents arguments.

Group	Event	Description	ID
Web Application	Site Collection Add	A site collection has been added.	800
	Site Collection Backup	A site collection has been backed up.	803
Site Collection	User Add	A user has been added.	300
	Group Add	A group has been added.	500
	Feature Update	A site collection or site feature has been activated or deactivated.	4400

Group	Event	Description	ID
	Social Follow Add	Followed users and tags on a user's profile news feed has been replicated.	7100
Web Site	Web Add	A web site has been added.	700
	Web File Add	A web site file has been added.	703
	Web Permission Add	A web permission has been added.	600
	Web Permission Level Add	A web permission level has been added.	400
	Content Type Add	A content type has been added.	3100
	Site Column Add	A site column has been added.	3200
	Feature Update	A site collection or site feature has been activated or deactivated.	4400
	Social Data Add	Social data has been added.	7000
	Site Navigation Update	Site navigation has been updated.	9010
	Site Master Page Update	A site master page has been updated.	9015
	Site Title Update	A site title has been updated.	9016
	Site Welcome Page Update	A site welcome page has been updated.	9017

Group	Event	Description	ID
	Site Tree View Update	A site tree view has been updated.	9018
	Site Theme Update	A site theme has been updated.	9019
	Site Logo Update	A site logo has been updated.	9020
List	List Add	A list has been added.	100
	View Add	A view has been added.	9000
List Item	List Item Add	A list item has been added.	0
	Folder Add	A folder has been added.	5
	Web Part Page Update	The web part page has been changed.	9030
Workflow	Workflow Association Add	A workflow association has been added.	4000
	Workflow Instance Add	A workflow instance has been added.	4102
Alerts	Alert Add	An alert has been added.	4300
Term Store (Managed Metadata)	Term Store Add	A Term Store has been added.	6000
	Navigation Term Store Add	A Navigation Term has been added.	6010

Note

Not all events will result in the generation of packages.

Backup Mode for Schedule Replicator Map

There are a few considerations to note with regards to BackupMode command for Schedule-ReplicatorMap.

Schedule-ReplicatorMap –BackupMode

The "Schedule-ReplicatorMap –BackupMode" option is equivalent to running the Backup-SPSite Powershell command. It backs up the local site collections' content, transports it, and then restores it on the remote farms.

Uses for BackupMode

BackupMode can be used in a DR environment when Replicator is enabled. Using this command and its frequency depends on a lot of factors like the size of the content, how often content changes, how up-to-date you want the DR site, etc.

This approach requires the following:

- The DR farm must be the same or a newer version of SharePoint.
- The source and target farms share the same Active Directory domain.
- The target site collection is an exact match of the source. i.e., you're not skipping events with rules or changing permissions on the target.
- One way replication from the primary site collection to the DR site collection. This command completely overwrites the DR site collection with the contents and settings from the primary one.
- Per Microsoft, the recommended maximum size for a site collection backup is 15 GB in SharePoint 2007 and 85 GB in SharePoint 2010.
- Per Microsoft, the site collection will be set to read-only for the duration of the backup to reduce the potential for user activity during the backup operation to corrupt the backup.

Start Replicator Processing

Start-ReplicatorProcessing

PowerShell	Start-ReplicatorProcessing	
Purpose	Force replication of a specified map family	
Edition	All	
Comment	The Map Family must also be set for Manual Replication.	
	URL	Enter a URL to a Web Application
	MapFamily	Enter a name for the Map Family.

InputMapFamily	Enter a name for the Map Family.

PowerShell Code Example

```
Start-ReplicatorProcessing -URL "http://localwebapplication" -MapFamily
"MapFamilyName"
```

Returns

None

Arguments

```
[[-URL] <String>]
[[-MapFamily] <String>]
[-InputMapFamily <ReplicationMapFamily>]
```

Compare Replicator Data

Compare-ReplicatorData

PowerShell	Compare-ReplicatorData
Purpose	Generate a report showing differences in list items and folders between the source and target lists. This command can also queue any folders and list items from the current web application to its target to synchronize the lists.
Edition	All
Comment	The report is also available from the SharePoint Site Settings page, under Replication Map settings.

URL	Enter a URL to a Web Application, Site Collection or a Web Site.
IncludeChildren	Include child web sites under the specified URL. If this parameter is not specified, child web sites are not included. This should not be used along with ListFolder.
MapFamily	The name of a Map Family to compare against. If you do not specify a Map Family, then Replicator will use a Map Family matching the URL.
Type	Specify whether the included URL is to a Web Application, Site Collection or a Web Site. If this parameter is missing, the URL will be interpreted as a Web Site.

Connection	The name of a connection specifying the target. If you do not specify a connection, then all connections included in the Map Family will be used.
QueueSetName - - - -	Tag all events queued with the specified name.
IgnoreConflicts	Ignore possible conflicts at the destination. (optional)
QueueEvents	Queue the items necessary to synchronize the lists.
UseCachedData	Use existing cached remote data.
ShowRemoteItems	Queue deletes for items that only exist on the other site.
ListTitle	The title for a specific list to compare.
ListFolder	The URL, relative to the List, to be used for filtering comparisons. The ListTitle argument is required for this to function. It should not to be used with IncludeChildren (see above).
OnlySpecifiedFolder	Specify whether subfolders inside the folder defined by ListFolder are visited for queuing. The ListFolder argument is required for this to function.
Verbose	Output more detailed information.
ForceMatchMode - - -	Match list items on both the source and target, with the same title, that have not been replicated to the target.
OnlyKeys - - - - - - -	Include only the Replicator keys when queuing.

PowerShell Code Example

```
Compare-ReplicatorData -URL "http://londonoffice" -MapFamily "Portal Map
Family" -IncludeChildren -ListTitle "Calendar"
```

Returns

None

Arguments

```
[-URL] <String>
[[-IncludeChildren] [<SwitchParameter>]]
-Type <String>
[-QueueEvents [<SwitchParameter>]]
[-ShowRemoteItems [<SwitchParameter>]]
[-UseCachedData [<SwitchParameter>]]
[-ListTitle <String>]
[-ListFolder <String>]
[-OnlySpecifiedFolder [<SwitchParameter>]]
```

```
[-Connection <String>]
[-MapFamily <String>]
[-OnlyKeys [<SwitchParameter>]]
[-ForceMatchMode [<SwitchParameter>]]
[-QueueSetName <String>]
[-IgnoreConflicts [<SwitchParameter>]]
```

Detailed Examples

To better understand Replication Reports, we suggest using the "-IncludeChildren" and "-ListTitle" arguments the first time you generate a report. This limits the scope of the command, making it easier to understand its results.

The following shows a sample Replication Report run on the Calendar list from http://londonoffice. We see that the "London Branch Party" is missing from the Corporate Portal (identified by the connection name).

Compare-ReplicatorData -URL "http://londonoffice"
-mapFamily "Portal Map Family" -includeChildren -listTitle "Calendar"
Operation started at 02/27/2013 4:17:59 PM.
Items marked as *** are not in sync with the Target Web Application.
Items marked as +++ only exist locally.
Items marked as --- do not exist locally.
Replication Map: Portal Map Family - / - Web Site
Connection: London to Corporate
Calendar
*** London Branch Party
Operation completed at 02/27/2013 4:18:19 PM.
The following example is run on a root site with two subsites. The report identifies that the /hr subsite is missing an announcement, two documents, and a folder.

Compare-ReplicatoData -URL http://corporateoffice
-mapFamily "Portal Map Family" -includeChildren
Operation started at 02/27/2013 10:27:30 AM.
Items marked as *** are not in sync with the Target Web Application.
Items marked as +++ only exist locally.
Items marked as --- do not exist locally.
Replication Map: Portal Map Family - / - Web Site
Connection: Corporate to London
Calendar
Shared Documents
Replication Map: Portal Map Family - /hr - Web Site
Connection: Corporate to London
Announcements
+++ Benefits Registration Deadline
HR Documents
+++ /Archives
+++ Dental Plan.pdf
+++ Holiday Schedule 2010.pdf
Replication Map: Portal Map Family - /public - Web Site
Connection: Corporate to London
Calendar
Documents

Master Page Gallery
Operation completed at 02/27/2013 10:27:54 AM.
In the preceding examples, the reports only show the unsynchronized content. By running the same command and adding the -QueueEvents argument, the command will also queue all items marked with *** and +++ to the target site.

Monitor Replication Networks

The following are commands used for Monitoring replication networks.

Write Replicator Info

Write-ReplicatorInfo

PowerShell	Write-ReplicatorInfo	
Purpose	Run support program to assist Metalogix Support.	
Edition	All	
Comment	Gets the Replicator information and saves it to the path specified.	
	ExportPath	Directory where created XML file will be saved.
	IncludeEvents	Include the Replicated Events data.
	NumberOfDays	Number of days for the Replicated Events data.
	SupportTicket	Metalogix Support ticket number. This argument automatically uploads the log file to Metalogix.

PowerShell Code Example

```
Write-ReplicatorInfo -ExportPath "C:\Temp"
```

Returns

None

Arguments

```
[-ExportPath <String>]
[-IncludeEvents [<SwitchParameter>]]
[-NumberOfDays <Int32>]
[-SupportTicket <String>]
```

Write Replicator Counters

Write-ReplicatorCounters

PowerShell	Write-ReplicatorCounters	
Purpose	Gets the current Replication Snap Counters.	
Edition	All	
Comment	Displays a list of the current Replication snap counters for the specified Web Application.	
	URL	Enter a URL to a Web Application.
	Connection	Enter a name for the Connection.
	Type	Snap Type: Current, Daily, Weekly, Custom.

PowerShell Code Example

```
Write-ReplicatorCounters -URL "http://localwebapplication"
```

Returns

```
Connection, CounterType, TotalPackagesProcessedInbound,
TotalItemsProcessedInbound, TotalOriginalSizeProcessedInbound,
TotalCompressedSizeProcessedInbound, TotalPackagesProcessedOutbound,
TotalItemsProcessedOutbound, TotalOriginalSizeProcessedOutbound,
TotalCompressedSizeProcessedOutbound, CreatedDate, LastModifiedDate
me:8080, Current, 0, 0, 0, 0, 0, 0, 0, 0, 1/1/0001 12:00:00 AM, 1/1/0001
12:00:00 AM
me:8080, HoURLy, 0, 0, 0, 0, 0, 0, 0, 0, 2/9/2009 7:11:17 PM, 2/12/2009
3:14:08 PM
me:8080, Daily, 0, 0, 0, 0, 0, 0, 0, 0, 2/9/2009 7:11:17 PM, 2/12/2009
12:53:14 PM
me:8080, Custom, 0, 0, 0, 0, 0, 0, 0, 0, 1/1/0001 12:00:00 AM, 1/1/0001
12:00:00 AM
```

Arguments

```
[-URL] <String>
[-Connection <String>]
[-Type <String>]
```

Write Replicator List Mappings

Write-ReplicatorListMappings

PowerShell	Write-ReplicatorListMappings	
Purpose	Get the lists remapped.	
Edition	Enterprise and Standard only	
Comment	Used on the Source Web Application only	
	URL	Enter a URL to a Web Application, Site Collection or a Web Site.
	Type	Specify whether the URL is to a Web Application, Site Collection or a Web Site. If this parameter is missing, the URL will be interpreted as a Web Site.
	MapFamily	Only add list mappings for the Replication Map from the specified Map Family.

PowerShell Code Example

```
Write-ReplicatorListMappings -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-MapFamily] <String>
[-Type <String>]
```

Write Replicator User Mappings

Write-ReplicatorUserMappings

PowerShell	Write-ReplicatorUserMappings
Purpose	Get user Mappings by Web Application or individual user and save them to an XML file.
Edition	Standard, Enterprise, and Runtime only

Comment	Gets all user Mappings from the specified Web Application and outputs them to an XML file called Usermapping.XML in the current directory.	
	URL	Enter a URL to a Web Application.
	GetAll	Get all user Mappings from the specified Web Application and save them to an XML file specified by SavePath parameter.
	GetUser	Get a specific user's Mapping from the specified Web Application and display to console.
	SavePath	The path of the XML file containing the user Mappings to be saved. Entering only a filename saves to the current directory while entering a full path saves to the specified location, which if doesn't exist, will be created.

PowerShell Code Example

```
Write-ReplicatorUserMappings -URL "http://localwebapplication" -GetAll
```

Returns

None

Arguments

```
[-URL] <String>
[-GetAll [<SwitchParameter>]]
[-GetUser <String>]
[-SavePath <String>]
```

Write Replicator Supported WSS Versions

Write-ReplicatorSupportedWSSVersions

PowerShell	Write-ReplicatorSupportedWSSVersions
Purpose	Get a list of supported WSS versions.
Edition	All
Comment	Writes the list of WSS versions supported by Replicator to the screen.

PowerShell Code Example

```
Write-ReplicatorSupportedWSSVersions
```

Returns

None

Arguments

None

Test Replicator List Consistency

Test-ReplicatorListConsistency

PowerShell	Test-ReplicatorListConsistency
Purpose	Tests the specified lists for consistency.
Edition	Standard, Enterprise, and Runtime only
Comment	Tests the specified lists or consistency.

URL	Enter a URL to a Web Application, Site Collection or a Web Site.
Type	Specify whether the URL is to a Web Application, Site Collection or a Web Site.
IncludeChildren	Fix child Web Site lists.
Fix	Fix lists on the specified sites. -If this is not included the names of the broken lists will be displayed.
ListTitle	Fix only the specified list.

PowerShell Code Example

```
Test-ReplicatorListConsistency -URL "http://localwebapplication" -
IncludeChildren -Fix
```

Returns

None

Arguments

```
[-URL] <String>
[-Type <String>]
[-IncludeChildren [<SwitchParameter>]]
[-Force [<SwitchParameter>]]
```

```
[-ListTitle <String>]
[-Fix [<SwitchParameter>]]
```

Write Replicator Supported Web Parts

Write-ReplicatorSupportedWebParts

PowerShell	Write-ReplicatorSupportedWebParts
Purpose	List Supported Web Parts.
Edition	All
Comment	None

PowerShell Code Example

```
Write-ReplicatorSupportedWebParts
```

Returns

Title	Short Title of the Web Part.
Type	Full namespace and class name for the Web Part.
Author	Company/Author who built the Web Part.
Assembly	.NET Assembly where the PersistWebPartPropertiesClass and ApplyWebPartPropertiesClass will be run.
PersistWebPartPropertiesClass	Class that will be run on the Outbound Farm to load Web Part Properties.
ApplyWebPartPropertiesClass	Class that will be run on the Inbound Farm to set Web Part Properties.

Arguments

None

Save Replicator Current Counters

Save-ReplicatorCurrentCounters

PowerShell	Save-ReplicatorCurrentCounters	
Purpose	Creates a new custom Counter.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application.
	Connection	Enter a name for the Connection.

PowerShell Code Example

```
Save-ReplicatorCurrentCounters -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
[-Connection <String>]
```

Maintain Replication Networks

The following are commands used for maintaining replication networks.

Register Offline Activation

Register-ReplicatorLicenseOffline

PowerShell	Register-ReplicatorLicenseOffline
Purpose	Generates and loads offline activation data.
Edition	All
Comment	

LicenseKey	License key issued by Metalogix.
ExportActivationDataFile	Creates a file with the activation data you can upload to the license server.
DisplayActivationDataText	Displays the encrypted activation data you can provide to the license server.
DisplayClearText	Displays the unencrypted activation data that is sent when licensing Replicator.
ImportActivationKeyFile	Imports the activation key file that was generated by the license server.

PowerShell Code Examples

```
Register-ReplicatorLicenseOffline -LicenseKey 11111-22222-33333-44444-55555
-ExportActivationDataFile "C:\activate.lic"

Register-ReplicatorLicenseOffline -LicenseKey 11111-22222-33333-44444-55555
-ImportActivationKeyFile "C:\Downloads\LicenseActivationResponse.dat"
```

Returns

None

Arguments

```
[-LicenseKey] <String>
[-ExportActivationDataFile <String>]
[-DisplayActivationDataText [<SwitchParameter>]]
[-DisplayClearText [<SwitchParameter>]]
[-ImportActivationKeyFile <String>]
```

Add Replicator Supported Web Part

Add-ReplicatorSupportedWebPart

PowerShell	Add-ReplicatorSupportedWebPart	
Purpose	Add a supported web part.	
Edition	All	
Comment	Adds specified name of a web part to the support web parts list maintained by Presentation Publisher. Web parts not on this list will not be replicated.	
	Title	Short title of the Web Part.
	Type	Full namespace and class of the Web Part.

Author	Company/Author who built the Web Part.
Assembly	.NET Assembly where the PersistWebPartPropertiesClass and ApplyWebPartPropertiesClass will be run.
PersistWebPartPropertiesClass	Class that will get run on the Outbound Farm to load Web Part Properties.
ApplyWebPartPropertiesClass	Class that will be run on the Inbound Farm to set Web Part Properties.

PowerShell Code Example

```
Add-ReplicatorSupportedWebPart -Title "XmlWebPart" -Type
Microsoft.SharePoint.WebPartPages.XmlWebPart -Author Microsoft
```

Returns

None

Arguments

```
[-Title] <String>
[-Type] <String>
[-Author <String>]
[-Assembly <String>]
[-PersistWebPartPropertiesClass <String>]
[-ApplyWebPartPropertiesClass <String>]
```

Add Replicator Supported WSS Versions

Add-ReplicatorSupportedWSSVersions

PowerShell	Add-ReplicatorSupportedWSSVersions	
Purpose	Deletes current list of all supported WSS versions and adds new list from XML file.	
Edition	All	
Comment	Updates the existing list of supported WSS versions with the one specified.	
	Filename	The full path of the XML file containing supported WSS versions.
	OverwriteVersions	Overwrite current list of WSS versions.

PowerShell Code Example

```
Add-ReplicatorSupportedWSSVersions -FileName "C:\SupportedWSSVersions.xml"
-OverwriteVersions
```

Returns

None

Arguments

```
[-Filename] <String>
[-OverwriteVersions [<Boolean>]]
```

Set Replicator Alerts

Set-ReplicatorAlerts

PowerShell	Set-ReplicatorAlerts	
Purpose	Enables, disables, or purges all alerts from the specified web application.	
Edition	Standard, Enterprise, and Runtime only	
Comment	This command must be run on the SharePoint farm where you want the action to occur.	
	URL	Enter a URL to a web application or a site collection.
	Action	Specify the action for the operation.
	Type	Specify if the action applies to all alerts in this web application or for a specific site collection.

PowerShell Code Example

```
Set-ReplicatorAlerts -URL "http://localwebapplication" -Action Enable
```

Returns

None

Arguments

```
[-URL] <String>
[-Action] <String>
[-Type <String>]
```

Remove Replicator Orphaned Config

Remove-ReplicatorOrphanedConfig

PowerShell	Remove-ReplicatorOrphanedConfig
Purpose	Clean out configuration for Web Applications that were previously deleted.
Edition	All
Comment	Removes configuration for web applications that no longer exist from the Replicator database.

PowerShell Code Example

```
Remove-ReplicatorOrphanedConfig
```

Returns

None

Arguments

None

Remove Replicator Orphaned Disk Objects

Remove-ReplicatorOrphanedDiskObjects

PowerShell	Remove-ReplicatorOrphanedDiskObjects	
Purpose	Deletes files from the Export path that are no longer referenced by Replication Packages.	
Edition	All	
Comment	Removes all files from Replicator's Export folder that are older than the oldest outbound queue item.	
	Quick	If this parameter is present, only files older than the oldest Package will be removed. If the parameter is not present, the file will be deleted if the matching Package is no longer in the database.

PowerShell Code Example

```
Remove-ReplicatorOrphanedDiskObjects -Quick
```

Returns

None

Arguments

```
[-Quick [<SwitchParameter>]]
```

Clear Replicator Web Application Configuration

Clear-ReplicatorWebAppConfig

PowerShell	Clear-ReplicatorWebAppConfig	
Purpose	Clears configuration for the specified Web Application.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application.
	IncludeReplicator	Delete Replicator database associated with this Web Application.
	RemoveReplicatorConfigDB	Delete Replicator configuration database.
	ContentDatabaseName	Remove the Replicator configuration from this database only.
	ForceContentDatabase	Force removal of Replicator configuration regardless of existing map families.
	RemoveFeatures	Remove Replicator Site Collection features.

PowerShell Code Example

```
Clear-ReplicatorWebAppConfig -URL "http://corporateoffice"
```

Returns

None

Arguments

```
[-URL] <String>
[-IncludeReplicator [<SwitchParameter>]]
[-RemoveReplicatorConfigDB [<SwitchParameter>]]
[-ContentDatabaseName <String>]
[-ForceContentDatabase [<SwitchParameter>]]
[-RemoveFeatures <String>]
```

Copy Replicator Alerts

Copy-ReplicatorAlerts

PowerShell	Copy-ReplicatorAlerts	
Purpose	Copies all alerts on all under the specified URL to target sites on other web applications. The copied alerts are kept on the source site, but can be disabled.	
Edition	Standard, Enterprise, and Runtime only	
Comment	Copies all alerts on all sites in the specified Web Application to mirroring sites on other Web Applications connected by Replication.	
	URL	Enter a URL to a web application or site collection. Alerts for this URL and any child sites will be copied.
	Type	Specify whether the URL is to a web application or a site collection. The default value is site collection.
	IncludeChildren	Include child web sites.
	Connection	Only copy alerts to the target of the specified connection.
	DisableInbound	Disable alerts on the target web application.
	DisableOutbound	Disable alerts on the source web application after copying them.
	UserName	Enter a user account to only move alerts for that account. Otherwise, alerts for all users will be copied.

PowerShell Code Example

```
Copy-ReplicatorAlerts -URL "http://localwebapplication" -IncludeChildren -
DisableInbound
```

Returns

None

Arguments

```
[-URL] <String>
[-IncludeChildren [<SwitchParameter>]]
[-DisableOutbound [<SwitchParameter>]]
[-DisableInbound [<SwitchParameter>]]
[-UserName <String>]
[-Connection <String>]
[-Type <String>]
```

Remove Replicator Supported Web Part

Remove-ReplicatorSupportedWebPart

PowerShell	Remove-ReplicatorSupportedWebPart	
Purpose	Delete Supported Web Part.	
Edition	All	
Comment		
	Type	Full namespace and class name for the Web Part.

PowerShell Code Example

```
Remove-ReplicatorSupportedWebPart -Type
Microsoft.SharePoint.WebPartPages.XmlWebPart
```

Returns

None

Arguments

```
[-Type] <String>
```

Export Replicator Settings

Export-ReplicatorSettings

PowerShell	Export-ReplicatorSettings
Purpose	Exports Replicator settings to a configuration file.
Edition	All
Comment	Passwords are not exported. You must add any necessary passwords to the exported settings file before you can import the settings.

FileName	The file to export the configuration settings to. If you do not specify a file, then settings are exported to ReplicatorSettings.xml in the current folder.
AppendToExisting	Add the exported settings to an existing file.
OverwriteExisting	Replace an existing file with the exported settings. This is the default behavior.
CustomBoundOnly	Only exports Replication Maps that use custom binding.

PowerShell Code Example

```
Export-ReplicatorSettings -FileName "C:\Test.xml"
```

Returns

None

Arguments

```
[[-FileName] <String>]
[[-AppendToExisting] [<SwitchParameter>]]
[[-OverwriteExisting] [<SwitchParameter>]]
[[-CustomBoundOnly] [<SwitchParameter>]]
```

Import Replicator Settings

Import-ReplicatorSettings

PowerShell	Import-ReplicatorSettings
Purpose	Imports Replicator settings from a configuration file.

Edition	All
Comment	Passwords are not exported. You must add any necessary passwords to the exported settings file before you can import the settings. If you delete any settings from an exported settings file, then those settings will be replaced by their default values when the settings are imported.

FileName	The file to import the configuration settings from.
ProcessWebAppSettings	Configure web applications using the imported settings. This is enabled by default.
ProcessGroups	Configure Replication Groups using the imported settings. This is enabled by default.
ProcessConnections	Configure Replication Connections using the imported settings. This is enabled by default.
ProcessClassifications	Configure Replication Classifications using the imported settings. This is enabled by default.
ProcessMapFamilies	Configure Replication Map Families using the imported settings. This is enabled by default.
TestConnections	Validate connections as they are added. This is enabled by default.
UseFarmAsTemplate	Replace the specified farm name in the configuration file with the name of the current farm.
UseWebApplicationIDAsRemoteId	Uses the local WA ID as the remote ID during an import instead of generating a new one. This setting should only be used in environments where all connections are in the same local farm.

PowerShell Code Example

```
Import-ReplicatorSettings -FileName "D:\ReplicatorSettings.xml" -
ProcessWebAppSettings
```

Returns

None

Arguments

```
[[-FileName] <String>]
[[-ProcessWebAppSettings] [<SwitchParameter>]]
[[-ProcessConnections] [<SwitchParameter>]]
[[-ProcessMapFamilies] [<SwitchParameter>]]
[[-ProcessGroups] [<SwitchParameter>]]
[[-ProcessClassifications] [<SwitchParameter>]]
[[-UseFarmAsTemplate] <String>]
```

```
[[-TestConnections] [<SwitchParameter>]]
[[-UseWebApplicationIDAsRemoteId] [<SwitchParameter>]]
```

Export Replicator KPI Settings

Export-ReplicatorKPISettings

PowerShell	Export-ReplicatorKPISettings
Purpose	Exports Replicator KPI settings from a configuration file.
Edition	All
Comment	Export Replicator Application Key Performance Indicator and Monitoring settings.

Exporting the file using ExportReplicatorKPISettings and changing elements within the exported file, such as thresholds, names, and whether features are enabled, and then importing the file using ImportReplicatorKPISettings allows you to override current KPI settings.

FileName	Specify the name and location of the output file. Default is the current executing directory.
AppendToExisting	Specify whether or not to append to the existing file.
OverwriteExisting	Specify whether or not to overwrite the existing node for the KPI, if it exists.
EditableKPIOnly	Specify whether to export only editable KPIs

PowerShell Code Example

```
Export-ReplicatorKPISettings -FileName "C:\Test.xml"
```

Returns

None

Arguments

```
[[-FileName] <String>]
[[-AppendToExisting] [<SwitchParameter>]]
[[-OverwriteExisting] [<SwitchParameter>]]
[[-EditableKPIOnly] [<SwitchParameter>]]
```

Import Replicator KPI Settings

Import-ReplicatorKPISettings

PowerShell	Import-ReplicatorKPISettings
Purpose	Imports Replicator KPI settings from a configuration file.
Edition	All
Comment	Import Replicator Application Key Performance Indicator and Monitoring settings. Exporting the file using ExportReplicatorKPISettings and changing elements within the exported file, such as thresholds, names, and whether features are enabled, and then importing the file using ImportReplicatorKPISettings allows you to override current KPI settings.

FileName	Specify the name and location of the intput file.
SpecificEngineName	Specify the name of the Replication engine to apply the KPIs to.
RestoreDefaults	Restore default replicator KPI settings.

PowerShell Code Example

```
Import-ReplicatorKPISettings -FileName "C:\Test.xml"
```

Returns

None

Arguments

```
[[-FileName] <String>]
[[-SpecificEngineName] <String>]
[[-RestoreDefaults] [<SwitchParameter>]]
```

Export Replicator KPI Results

Export-ReplicatorKPIResults

PowerShell	Export-ReplicatorKPIResults
Purpose	Exports Replicator KPI results from a configuration file.

Edition	All	
Comment		
	FileName	Specify the name and location of the output file. Default is the current executing directory.
	KPISetName	Specify the name of the KPI set for which to export KPI results.

PowerShell Code Example

```
Export-ReplicatorKPIResults -FileName "C:\Test.xml"
```

Returns

None

Arguments

```
[[-FileName] <String>]
[-KPISetName] <String>
```

Import Replicator KPI Results

Import-ReplicatorKPIResults

PowerShell	Import-ReplicatorKPIResults	
Purpose	Imports Replicator KPI results from a configuration file.	
Edition	All	
Comment		
	FileName	Specify the name and location of the output file. Default is the current executing directory.
	Rename Result Name to Farm Name	Renames the results name to be the same as the farm name.

PowerShell Code Example

```
Import-ReplicatorKPIResults -FileName "C:\Test.xml"
```

Returns

None

Arguments

```
[[-FileName] <String>]
[[-RenameResultNameToFarmName] [<Boolean>]]
```

Clear Replicator Map Queue

Clear-ReplicatorMapQueue

PowerShell	Clear-ReplicatorMapQueue	
Purpose	Clear all queued Replication Packages for the specified Web Site.	
Edition	All	
Comment	Clears all queued Packages from the queue for the specified Web Site.	
	URL	Enter a URL to a Web Site.
	IncludeChildren	Clear the queued Replication Packages for the specified Web Site and any child Web Sites.
	Direction	Specify whether to target Inbound or Outbound Packages.

PowerShell Code Example

```
Clear-ReplicatorMapQueue -URL "http://localwebapplication" -IncludeChildren
 -Direction Outbound
```

Returns

None

Arguments

```
[-URL] <String>
[[-IncludeChildren] [<SwitchParameter>]]
[-Direction <String>]
```

Move Replicator Alerts

Move-ReplicatorAlerts

PowerShell	Move-ReplicatorAlerts
Purpose	Transfers all alerts on all under the specified URL to target sites on other web applications. The moved alerts are removed from the source site.
Edition	Standard, Enterprise, and Runtime only
Comment	

URL	Enter a URL to a web application or site collection. Alerts for this URL and any child sites will be moved.
Type	Specify whether the URL is to a web application or a site collection. The default value is site collection.
IncludeChildren	Move alerts for child sites under the URL.
Connection	Only move alerts to the target of the specified connection.
DisableInbound	Disable alerts on the target web application.
UserName	Enter a user account to only move alerts for that account. Otherwise, alerts for all users will be transferred.

PowerShell Code Example

```
Move-ReplicatorAlerts -URL "http://localwebapplication" -Includechildren -
DisableInbound -UserName "corporate\jdoe"
```

Returns

None

Arguments

```
[-URL] <String>
[-IncludeChildren [<SwitchParameter>]]
[-DisableInbound <String>]
[-UserName <String>]
[-Connection <String>]
[-Type <String>]
```

Clear Replicator Queue All

Clear-ReplicatorQueueAll

PowerShell	Clear-ReplicatorQueueAll
Purpose	Reset queues by clearing all Packages.
Edition	All
Comment	

URL	Enter a URL to a Web Application.
Direction	Specify whether to clear inbound or outbound packages, or both.
DeleteNow	Deletes everything in the Replicator queue immediately instead of scheduling for deletion.
CompleteThreshold	Clears all locally completed packages older than the specified Date (MM/DD/YYYY)

PowerShell Code Example

```
Clear-ReplicatorQueueAll -URL "http://corporateoffice" -Direction Outbound
```

Returns

None

Arguments

```
[-URL] <String>
[-Direction] <String>
[[-DeleteNow] <String>]
[-CompleteThreshold] <String>
```

Clear Replicator Queue Completed

Clear-ReplicatorQueueCompleted

PowerShell	Clear-ReplicatorQueueCompleted
Purpose	Clear Packages which have completed and do not contain any errors. Packages yet to be processed, packages that are currently being processed or

	Packages that contain conflicts will not be cleared.
Edition	All
Comment	

URL	Enter a URL to a Web Application.
DeleteNow	Deletes all completed items in the Replicator queue immediately, instead of scheduling for deletion.
CompleteThreshold	Clears all locally completed packages older than the specified Date (MM/DD/YYYY)

PowerShell Code Example

```
Clear-ReplicatorQueueCompleted -URL "http://corporateoffice"
```

Returns

None

Arguments

```
[-URL] <String>
[[-DeleteNow] <String>]
[-CompleteThreshold] <String>
```

Clear Replicator Queue Completed or Error

Clear-ReplicatorQueueCompletedOrError

PowerShell	Clear-ReplicatorQueueCompletedOrError
Purpose	Clears all Replication Packages with a status of Completed or Error. Packages waiting to be processed, packages that are currently being processed or packages that contain conflicts will not be cleared.
Edition	All
Comment	

URL	Enter a URL to a Web Application.
Direction	Specify whether to clear inbound or outbound packages, or both.
DeleteNow	Deletes all Completed or Error items in the Replicator queue immediately, instead of scheduling for deletion.

	CompleteThreshold	Clears all locally completed packages older than the specified Date (MM/DD/YYYY)

PowerShell Code Example

```
Clear-ReplicatorQueueCompletedOrError -URL "http://localwebapplication" -
Direction Outbound
```

Returns

None

Arguments

```
[-URL] <String>
[-Direction] <String>
[[-DeleteNow] <String>]
[-CompleteThreshold] <String>
```

Reapply Replicator Error Queue Items

Reapply-ReplicatorErrorQueueItems

PowerShell	Reapply-ReplicatorErrorQueueItems	
Purpose	Reapply inbound queue items that have errors.	
Edition	All	
Comment		
	URL	Enter a URL to a Web Application.

PowerShell Code Example

```
Reapply-ReplicatorErrorQueueItems -URL "http://corporateoffice"
```

Returns

None

Arguments

```
[-URL] <String>
```

Install-ReplicatorSolution

Install-ReplicatorSolution

PowerShell	Install-ReplicatorSolution
Purpose	Redeploys the Replicator WSS solutions.
Edition	All
Comment	None

PowerShell Code Example

```
Install-ReplicatorSolution
```

Returns

None

Arguments

None

Test Replicator Consistency

Test-ReplicatorConsistency

PowerShell	Test-ReplicatorConsistency	
Purpose	Checks the consistency of the Replicator databases.	
Edition	All	
Comment	This command checks the Replicator databases for consistency and outputs any errors that are found.	
	Fix	Corrects detected consistency issues.

PowerShell Code Example

```
Test-ReplicatorConsistency
```

Returns

None

Arguments

```
[-Fix [<SwitchParameter>]]
```

Start Replicator Maintenance

Start-ReplicatorMaintenance

PowerShell	Start-ReplicatorMaintenance	
Purpose	Run a maintenance cycle on the specified Web Application.	
Edition	All	
Comment	This command starts a maintenance cycle for the specified Web Application.	
	URL	Enter a URL to a Web Application.

PowerShell Code Example

```
Start-ReplicatorMaintenance -URL "http://localwebapplication"
```

Returns

None

Arguments

```
[-URL] <String>
```

Update Replicator Account

Update-ReplicatorAccount

PowerShell	Update-ReplicatorAccount
Purpose	Updates the local user accounts used by Replicator. This command updates the Replicator service account and IIS virtual directory account.
Edition	Standard, Enterprise, and Runtime only

Comment

ReplicatorServiceAccount	If specified, changes the user account and password for the Replicator service account.
VirtualDirectoryAccount	If specified, changes the user account and password for the Replicator virtual directories in IIS.
URL	The URL of the web application you are changing accounts for. If you do not specify a URL with VirtualDirectoryAccount, then this account will be used for all web applications where Replicator is enabled.
UserAccount	The existing user account.
Password	The password for the existing user account.
NewUserAccount	The new user account to use.
NewPassword	The new password to use.

NOTE: You must specify at least one of ReplicatorServiceAccount or VirtualDirectoryAccount.

PowerShell Code Example

```
Update-ReplicatorAccount -VirtualDirectoryAccount true -URL
"http://corporateoffice" -UserAccount "corporateoffice\spadmin" -Password
"p4ssw0rd" -NewPassword "pa55w0rd"
```

Returns

None

Arguments

```
[-UserAccount] <String>
[-Password] <String>
[-NewPassword] <String>
[-NewUserAccount <String>]
[-VirtualDirectoryAccount [<SwitchParameter>]]
[-ReplicatorServiceAccount [<SwitchParameter>]]
[-URL <String>]
```

Write Replicator Features

Write-ReplicatorFeatures

PowerShell	Write-ReplicatorFeatures

Purpose	Monitors what Replicator features are enabled and disabled.
Edition	All
Comment	None

PowerShell Code Example

```
Write-ReplicatorFeatures
```

Returns

None

Arguments

None

Update Replicator Settings File

Update-ReplicatorSettingsFile

PowerShell	Update-ReplicatorSettingsFile	
Purpose	Updates the existing Replicator settings file to use a new export/import structure.	
Edition	All	
Comment		
	InputFileName	Enter the file name for the input file.
	OutputFileName	Enter the file name for the output file.

PowerShell Code Example

```
Update-ReplicatorSettingsFile
```

Returns

None

Arguments

```
[[-InputFileName] <String>]
[[-OutputFileName] <String>]
```

Managing Administrative Farm Settings

The following are commands used for managing administrative farm settings.

Set Replicator Farm Name

Set-ReplicatorFarmName: Set the Replicator Farm Name

PowerShell	Set-ReplicatorFarmName	
Purpose	Sets the Replicator Farm Name.	
Edition	All	
Comment		
	Friendly Name	Enter a friendly farm name.

PowerShell Code Example

```
Set-ReplicatorFarmName -FriendlyName "FrienlyFarmName"
```

Returns

None

Arguments

```
[-Friendly Name] <String>
```

Write Replicator Farm Name

Write-ReplicatorFarmName: Write the Replicator Farm Name.

PowerShell	Write-ReplicatorFarmName
Purpose	This command will provide the Replicator Farm Name.
Edition	All

Comment	None

PowerShell Code Example

```
Write-ReplicatorFarmName Returns
```

Returns

None

Arguments

None

Remove Replicator Farm From Network Health

Remove-ReplicatorFarmFromNetworkHealth

PowerShell	Remove-ReplicatorFarmFromNetworkHealth	
Purpose	This command will remove the specified farm from Network Health.	
Edition	All	
Comment		
	Farm Name	Enter the farm name.

PowerShell Code Example

```
Remove-ReplicatorFarmFromNetworkHealth -FarmName "CorporateOffice"
```

Returns

None

Arguments

```
[[-FarmName] <String>]
```

Set Replicator Administrative Farm Settings

Set-ReplicatorAdministrativeFarmSettings

PowerShell	Set-ReplicatorAdministrativeFarmSettings	
Purpose	Sets the Replicator Administrative farm settings.	
Edition	All	
Comment		
	URL	Enter the URL of the Administration Web Application for Replicator.
	Name	Enter the farm name.

PowerShell Code Example

```
Set-ReplicatorAdministrativeFarmSettings -URL
"htt://WebApplicationURL:Port" -Name "Farm Name"
```

Returns

None

Arguments

```
[-URL] <String>
[[-Name] <String>]
```

Clear Replicator Administrative Farm Settings

Clear-ReplicatorAdministrativeFarmSettings

PowerShell	Clear-ReplicatorAdministrativeFarmSettings
Purpose	Clears the Replicator Administrative farm settings.
Edition	All
Comment	None

PowerShell Code Example

```
Clear-ReplicatorAdministrativeFarmSettings
```

Returns

None

Arguments

None

Write Replicator Administrative Farm Settings

Write-ReplicatorAdministrativeFarmSettings

PowerShell	Write-ReplicatorAdministrativeFarmSettings
Purpose	Lists the Replicator Administrative farm settings.
Edition	All
Comment	None

PowerShell Code Example

```
Write-ReplicatorAdministrativeFarmSettings
```

Returns

None

Arguments

None

Queue Replicator Administrative Farm Settings

Queue-ReplicatorAdministrativeFarmSettings

PowerShell	Queue-ReplicatorAdministrativeFarmSettings
Purpose	Queue any changes to the administrative settings of the Farm specified.

Edition	All
Comment	None

PowerShell Code Example

```
Queue-ReplicatorAdministrativeFarmSettings
```

Returns

None

Arguments

None

Add Replicator Administrative Alerts

Add-ReplicatorAdministrativeAlerts

PowerShell	Add-ReplicatorAdministrativeAlerts
Purpose	Adds an email address to the list of emails where Replicator Administrative Alerts are sent.
Edition	All
Comment	

EmailAddress	Add the specified email address to the list of emails where administrative alerts will be sent.

PowerShell Code Example

```
Add-ReplicatorAdministrativeAlerts -EmailAddress "UserName@ServerMail.com"
```

Returns

None

Arguments

```
-EmailAddress <String>
```

Remove Replicator Administrative Alerts

Remove-ReplicatorAdministrativeAlerts

PowerShell	Remove-ReplicatorAdministrativeAlerts	
Purpose	Removes emails from the list of emails where Replicator Administrative Alerts are sent.	
Edition	All	
Comment		
	EmailAddress	Removes the email address from the list of emails where administrative alerts are sent.

PowerShell Code Example

```
Remove-ReplicatorAdministrativeAlerts -EmailAddress
"UserName@ServerMail.com"
```

Returns

None

Arguments

```
-EmailAddresss <String>
```

Write Replicator Administrative Alerts

Write-ReplicatorAdministrativeAlerts

PowerShell	Write-ReplicatorAdministrativeAlerts
Purpose	Lists email addresses where Replicator Administrative Alerts are sent.
Edition	All
Comment	None

PowerShell Code Example

```
Write-ReplicatorAdministrativeAlerts
```

Returns

None

Arguments

None

5

Appendix – Repadm equivalents for PowerShell commands

The following appendix provides an overview of how to use repadm with Replicator. Please note that we strongly suggest that you switch over to PowerShell, as all updates to the product will occur for PowerShell commands. Repadm commands will no longer be updated.

Repadm Commands

The repadm command is similar in structure to the SharePoint Stsadmin command line tool. Repadm.exe is located in the main Replicator installation folder. The default installation folder is one of the following, depending on which version of SharePoint you are using:

- C:\Program Files\Metalogix\Replicator\WSS40
- C:\Program Files\Metalogix\Replicator\WSS30

Run repadm from a Command Prompt window on the server where Replicator is installed. If UAC is enabled on this server, then you must run the Command Prompt as an administrator.

The following code example shows how to run enable a web application using repadm:

```
Repadm -o ConfigureWebApp   -URL http://corporateoffice -Enabled true
```

NOTE: If an argument value contains a space, then you must surround the entire argument value with double quotes. For example, -MapFamily "Portal Map Family".

Running Repadm Commands from PowerShell

The following code example shows how to run repadm commands from PowerShell:

```
'C:\Program Files\Metalogix\Replicator\WSS40\repadm.exe' -o
ConfigureWebApplication -URL http://corporateoffice -Enabled true
```

The following table will provide the Repadm to PowerShell equivalents so that you can seamlessly switch over to PowerShell:

Repadm	PowerShell
ActivateOffline	Register-ReplicatorLicenseOffline
AddConnectionBinding	Add-ReplicatorConnectionBinding
AddListMappings	Add-ReplicatorListMappings

AddMapFamily	Add-ReplicatorMapFamily
AddReplicationGroup	Add-ReplicatorGroup
AddReplicationGroupConnection	Add-ReplicatorGroupConnection
AddSupportedWebPart	Add-ReplicatorSupportedWebPart
AddUserMappings	Add-ReplicatorUserMappings
AddWSSVersions	Add-ReplicatorSupportedWSSVersions
Alerts	Set-ReplicatorAlerts
CleanOrphanedConfig	Remove-ReplicatorOrphanedConfig
CleanOrphanedDiskObjects	Remove-ReplicatorOrphanedDiskObject
ClearConfiguration	Clear-ReplicatorWebAppConfig
ConfigureWebApplication	Set-ReplicatorWebAppConfig
CopyAlerts	Copy-ReplicatorAlerts
DeleteListMappings	Remove-ReplicatorListMappings
DeleteMapFamily	Remove-ReplicatorMapFamily
DeleteReplicationGroup	Remove-ReplicatorGroup
DeleteReplicationGroupConnection	Remove-ReplicatorGroupConnection
DeleteSupportedWebPart	Remove-ReplicatorSupportedWebPart
DeleteUserMappings	Remove-ReplicatorUserMappings
DisableAlertReplication	Disable-ReplicatorAlertReplication
DisableMap	Disable-ReplicatorMap
DisableMapFamily	Disable-ReplicatorMapFamily
DumpInfo	Write-ReplicatorInfo
EnableAlertReplication	Enable-ReplicatorAlertReplication
EnableMap	Enable-ReplicatorMap
EnableMapFamily	Enable-ReplicatorMapFamily
ExportReplicatorKPISettings	Export-ReplicatorKPISettings
ExportReplicatorSettings	Export-ReplicatorSettings
GetCounters	Write-ReplicatorCounters
GetListMappings	Write-ReplicatorListMappings
GetUserMappings	Write-ReplicatorUserMappings
GetWSSVersions	Write-ReplicatorSupportWSSVersions
ImportReplicatorKPISettings	Import-ReplicatorKPISettings
ImportReplicatorSettings	Import-ReplicatorSettings

ListConsistencyCheck	Test-ReplicatorListConsistency
ListDefaultReplicatorDataFolder	Write-ReplicatorDefaultDataFolder
ListMap	Write-ReplicatorMap
ListMapFamilies	Write-ReplicatorMapFamilies
ListMaps	Write-ReplicatorMaps
ListReplicatorDataFolders	Write-ReplicatorDataFolders
ListSupportedWebParts	Write-ReplicatorSupportedWebParts
MapClearQueue	Clear-ReplicatorMapQueue
MoveAlerts	Move-ReplicatorAlerts
N/A	Get-ReplicatorWebApp
N/A	Get-ReplicatorGroup
N/A	Get-ReplicatorMapFamily
N/A	Get-ReplicatorMap
N/A	Get-ReplicatorConnections
OfflineExport	Export-ReplicatorPackages
OfflineImport	Import-ReplicatorPackages
QueueMap	Queue-ReplicatorMap
QueuesClearAll	Clear-ReplicatorQueueAll
QueuesClearCompleted	Clear-ReplicatorQueueCompleted
QueuesClearCompletedOrError	Clear-ReplicatorQueueCompletedOrError
RDCCache	Set-ReplicatorRDCCache
RDCDynamicCache	Set-ReplicatorRDCDynamicCache
ReapplyErrorQueueItems	Reapply-ReplicatorErrorQueueItems
RecordCounters	Save-ReplicatorCurrentCounters
RedeploySolutions	Install-ReplicatorSolution
RefreshConnections	Update-ReplicatorConnections
RepConsistencyCheck	Test-ReplicatorConsistency
ReplicateMapFamily	Start-ReplicatorProcessing
ReplicationReport	Compare-ReplicatorData
ResetMap	Set-ReplicatorMapInheritance
ResetMapFamily	Set-ReplicatorMapFamily
RunMaintenance	Start-ReplicatorMaintenance
UpdateConnection	Update-ReplicatorConnection

UpdateDefaultReplicatorDataFolder	Update-ReplicatorDefaultDataFolder
UpdateMap	Update-ReplicatorMap
UpdateMapConnection	Update-ReplicatorMapConnection
UpdateMapFamily	Update-ReplicatorMapFamily
UpdateMapLists	Update-ReplicatorMapLists
UpdateReplicationGroup	Update-ReplicatorGroup
UpdateReplicatorAccount	Update-ReplicatorAccount

6

About

We are more than just a name. We are on a quest to make your information technology work harder for you. That is why we build community-driven software solutions that help you spend less time on IT administration and more time on business innovation. We help you modernize your data center, get you to the cloud quicker and provide the expertise, security and accessibility you need to grow your data-driven business. Combined with Quest's invitation to the global community to be a part of its innovation, and our firm commitment to ensuring customer satisfaction, we continue to deliver solutions that have a real impact on our customers today and leave a legacy we are proud of. We are challenging the status quo by transforming into a new software company. And as your partner, we work tirelessly to make sure your information technology is designed for you and by you. This is our mission, and we are in this together. Welcome to a new Quest. You are invited to Join the Innovation™.

Our brand, our vision. Together.

Our logo reflects our story: innovation, community and support. An important part of this story begins with the letter Q. It is a perfect circle, representing our commitment to technological precision and strength. The space in the Q itself symbolizes our need to add the missing piece — you — to the community, to the new Quest.

Contact Quest

For sales or other inquiries, visit www.quest.com/contact.

Technical Support Resources

Technical support is available to Quest customers with a valid maintenance contract and customers wh o 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