The Powershell Book Generator

PowerShell with AppX

The PowerShell Book Genrator - Andreas Nick 2019

http://www.andreasnick.com

http://www.software-virtualisierung.de

Table Of Contents

Cmdlet: Add-AppxPackage

Synops

Adds a signed app package to a user account.

Syntax

```
Add-AppxPackage [-DependencyPackages <String[]>]
    [-ForceApplicationShutdown] [-ForceTargetApplicationShutdown]

    [-InstallAllResources] -MainPackage <String> [-Register] [-
Confirm]
    [-WhatIf] [<CommonParameters>]


Add-AppxPackage [-DependencyPackages <String[]>]
    [-ForceApplicationShutdown] [-ForceTargetApplicationShutdown]

    [-InstallAllResources] -MainPackage <String> [-
OptionalPackages
    <String[]>] -RegisterByFamilyName [-Confirm] [-WhatIf]
[<CommonParameters>]


Add-AppxPackage [-Path] <String> [-DependencyPath <String[]>]
    [-ExternalPackages <String[]>] [-ForceApplicationShutdown]
    [-ForceTargetApplicationShutdown] [-InstallAllResources]
    [-OptionalPackages <String[]>] [-RequiredContentGroupOnly] [-
Volume
    <AppxVolume>] [-Confirm] [-WhatIf] [<CommonParameters>]


Add-AppxPackage [-Path] <String> [-DependencyPath <String[]>]
    [-DisableDevelopmentMode] [-ForceApplicationShutdown]
    [-ForceTargetApplicationShutdown] [-InstallAllResources] -
Register
    [-Confirm] [-WhatIf] [<CommonParameters>]


Add-AppxPackage [-Path] <String> [-DependencyPath <String[]>]
    [-ForceApplicationShutdown] [-ForceTargetApplicationShutdown]
```

```
    [-InstallAllResources] [-RequiredContentGroupOnly] -Update [-
Confirm]
    [-WhatIf] [<CommonParameters>]
```

Description

The Add-AppxPackage cmdlet adds a signed app package to a user account. An app package has an .appx file name extension. Use the DependencyPath parameter to add all other packages that are required for the installation of the app package.

You can use the Register parameter to install from a folder of unpackaged files during development of Windows® Store apps.

To update an already installed package, the new package must have the same package family name.

Parameters

Parameter :**DependencyPackages**

Description :Specifies the dependency package full name or dependency package bundle full name to be registered.

Required	false
Position	named
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**DependencyPath**

Description :Specifies an array of file paths of dependency packages that are required for the installation of the app package. The app package has an .appx or .appxbundle file name extension. You can specify the paths to more than one dependency package.

Required	false
Position	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**DisableDevelopmentMode**

Description :Indicates that this cmdlet registers an existing app package installation that has been disabled, did not register, or has become corrupted. Use the current parameter to specify that the manifest is from an existing installation, and not from a collection of files in development mode. You can also use this parameter to register an application that the Package Manager API (http://go.microsoft.com/fwlink/?LinkId=245447)has staged. Use the Register parameter to specify the location of the app package manifest .xml file from the installation location.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**ExternalPackages**

Description :Specifies an array of optional packages that must be installed along with the app package. It is an atomic operation which means that if the app or its optional packages fail to install, the deployment operation will be aborted

Required	false
Position	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**ForceApplicationShutdown**

Description :Indicates that this cmdlet forces all active processes that are associated with the package or its dependencies to shut down. If you specify this parameter, do not specify the ForceTargetApplicationShutdown parameter.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**ForceTargetApplicationShutdown**

Description :Indicates that this cmdlet forces all active processes that are associated with the package to shut down. If you specify this parameter, do not specify the ForceApplicationShutdown parameter.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**InstallAllResources**

Description :Indicates that this cmdlet forces the deployment of all resource packages specified from a bundle argument. This overrides the resource applicability check of the deployment engine and forces staging of all resource packages, registration of all resource packages, or staging and registration of all resource packages. This parameter can only be used when specifying a resource bundle or resource bundle manifest.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**MainPackage**

Description :Specifies the main package full name or bundle full name to register.

Required	true
Position	named
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**OptionalPackages**

Description :Specifies the PackageFamilyName of the optional packages that are in a related set that need to be installed along with the app. Unlike the external packages flag, you do not need to pass in a path to the optional package(s). It is an atomic operation which means that if the app or its optional packages fail to install, the deployment operation will be aborted

Required	false
Position	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Path**

Description :Specifies the file path of the app package. An app package has an .appx or .appxbundle file name extension.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**Register**

Description :Indicates that this cmdlet registers an application in development mode. You can use development mode to install applications from a folder of unpackaged files. You can use the current parameter to test your Windows® Store apps before you deploy them as app packages. To register an existing app package installation, you must specify the DisableDevelopmentMode parameter and the Register parameter. In order to specify dependency packages, specify the DependencyPath parameter and the DisableDevelopmentMode parameter.

Required	true
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**RegisterByFamilyName**

Description :{{Fill RegisterByFamilyName Description}}

Required	true
Position	named
Default value	False

Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**RequiredContentGroupOnly**

Description :Specifies that only the required content group that is specified in the AppxContentGroupMap.xml must be installed. At this point the app can be launched. Calling add-appxpackage specifying the path to the app, triggers the rest of the app to be installed in the order defined in the AppxContentGroupMap.xml.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Update**

Description :Specifies that the package being added is a dependency package update. A dependency package is removed from the user account when the parent app is removed. If you do not use this parameter, the package being added is a primary package and is not removed from the user account if the parent app is removed. To update an already installed package, the new package must have the same package family name.

Required	true
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Volume**

Description :Specifies the AppxVolume object to which to stage the package. The volume also specifies the default location for user AppData .

Required	false
Position	named
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Inputs

Type : System.String[] System.IO.FileInfo

Outputs

Type : None

Notes

Examples

Example 1: Add an app package

This command adds an app package that the package contains.

```
PS C:\> Add-AppxPackage -Path
"C:\Users\user1\Desktop\MyApp.appx" -DependencyPath
"C:\Users\user1\Desktop\winjs.appx"
```

Example 2: Add a disabled app package in development mode

This command gets the full path of the package manifest file of an installed Windows

Store app, and then registers that package. You can use DisableDevelopmentMode to register an application that is staged by the StagePackageAsync API, has been disabled, or has become corrupted during testing.

```
PS C:\> $ManifestPath = (Get-AppxPackage -Name
"*WindowsCalculator*").InstallLocation + "\Appxmanifest.xml"
PS C:\> Add-AppxPackage -Path $ManifestPath -Register -
DisableDevelopmentMode
```

Example 3: Add an app along with its optional packages

This command adds an app package along with its optional packages. It is an atomic operation which means that if the app or its optional packages fail to install, the deployment operation will be aborted

```
PS C:\> Add-AppxPackage -Path
"C:\Users\user1\Desktop\MyApp.appxbundle" -ExternalPackages
"C:\Users\user1\Desktop\optionalpackage1.appx","C:\Users\user1
\Desktop\optionalpackage2.appxbundle"

PS C:\> Add-AppxPackage -Path
"C:\Users\user1\Desktop\MyApp.appxbundle" -OptionalPackages
"29270sandstorm.OptionalPackage1_gah1vdar1nn7a"
```

Example 4: Install only the required section of a streaming app

This command adds an app package but only installs the required section of a streaming app. Calling this command again without the RequiredContentGroupOnly flag proceeds to install the rest of the application in the order defined by the AppxContentGroupMap.xml

```
PS C:\> Add-AppxPackage -Path
"C:\Users\user1\Desktop\MyApp.appxbundle" -
RequiredContentGroupOnly
```

Cmdlet: Add-AppxVolume

Synops

Adds an appx volume to the Package Manager.

Syntax

```
Add-AppxVolume [-Path] <String[]> [-Confirm] [-WhatIf]
[<CommonParameters>]
```

Description

The Add-AppxVolume cmdlet adds an AppxVolume for the Package Manager to advertise. After you add a volume, appx deployment operations can use that volume as a target. This cmdlet returns the volume that it adds. Note, the Path parameter must be specified as a drive letter followed by "WindowsApps" as the directory. Not using the aforementioned format could lead to inconsistent behavior in the application model subsystems or the volume itself; for more information see the examples section.

Parameters

Parameter :**Path**

Description :Specifies the path of the mount point of the volume that this cmdlet adds.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False

Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Outputs

This cmdlet returns the AppxVolume object that it adds.

Type : Microsoft.Appx.PackageManager.Commands.AppxVolume

Notes

Cmdlet: Dismount-AppxVolume

Synops

Dismounts an appx volume.

Syntax

```
Dismount-AppxVolume [-Volume] <AppxVolume[]> [-Confirm] [-WhatIf]

    [<CommonParameters>]
```

Description

The Dismount-AppxVolume cmdlet dismounts an AppxVolume . After you dismount a volume, all apps that are deployed to that target become inaccessible.

Parameters

Parameter :**Volume**

Description :Specifies the AppxVolume object to dismount.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Notes

Examples

Example 1: Dismount a volume by using a path

This command dismounts a volume at the path E:\.

```
PS C:\> Dismount-AppxVolume -Volume E:\
```

Example 2: Dismount a volume by using an ID

This command dismounts a volume that has the specified media ID.

```
PS C:\> Dismount-AppxVolume -Volume {7e62a691-398e-4fbe-819a-
64f1e407777a}
```

Cmdlet: Get-AppxDefaultVolume

Synops

Gets the default appx volume.

Syntax

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

Description

The Get-AppxDefaultVolume cmdlet gets the default AppxVolume . The default AppxVolume is the default target for all deployment operations on the computer. You cannot remove the default AppxVolume from the list of volumes.

Parameters

Outputs

Type : Microsoft.Appx.PackageManager.Commands.AppxVolume

Notes

Cmdlet: Get-AppxLastError

Synops

Get the last error reported in the app package installation logs.

Syntax

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

Description

The Get-AppxLastError cmdlet gets the last error reported in the app package installation logs for the current Windows PowerShell® session. An app package has an .appx file name extension.

Parameters

Inputs

Type : None

Outputs

Type : System.Diagnostics.Eventing.Reader.EventLogRecord

Notes

Cmdlet: Get-AppxLog

Synops

Gets an app package installation log.

Syntax

```
Get-AppxLog [-ActivityId <String>] [<CommonParameters>]


Get-AppxLog [-All] [<CommonParameters>]
```

Description

The Get-AppxLog cmdlet gets the app package installation log created during the deployment of an app package. An app package has an .appx file name extension. The log contains errors, warnings, and additional information about the processes initiated by cmdlets in the Appx Windows PowerShell® module.

When Add-AppxPackage or Remove-AppxPackage report a failure, they return the ActivityID to use with Get-AppxLog .

For more information about common error codes, see Troubleshooting packaging, deployment, and query of Windows Store apps (http://go.microsoft.com/fwlink/?LinkId=271201).

Parameters

Parameter :**ActivityId**

Description :Specifies an activity ID. This cmdlet uses the ID to get the log for a particular app package installation.

Required	false
Position	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**All**

Description :Indicates that the cmdlet gets all logs on the computer. You can get

additional information when you run this cmdlets from Windows PowerShell as an administrator.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Inputs

Type : System.String[]

Outputs

Type : System.Diagnostics.Eventing.Reader.EventLogRecord

Notes

Examples

Example 1: Get logs for the most recent deployment

This command gets the logs associated with the most recent deployment operation.

```
PS C:\> Get-AppxLog
```

Example 2: Get logs for all logs

This command gets all the app package installation logs on the computer.

```
PS C:\> Get-AppxLog -All
```

Cmdlet: Get-AppxPackage

Synops

Gets a list of the app packages that are installed in a user profile.

Syntax

```
Get-AppxPackage [[-Name] <String>] [[-Publisher] <String>] [-
AllUsers]
    [-PackageTypeFilter {None | Main | Framework | Resource |
Bundle | Xap}]
    [-User <String>] [-Volume <AppxVolume>] [<CommonParameters>]
```

Description

The Get-AppxPackage cmdlet gets a list of the app packages that are installed in a user profile. An app package has an .appx file name extension. To get the list of packages for a user profile other than the profile for the current user, you must run this command by using administrator permissions.

Parameters

Parameter :**AllUsers**

Description :Indicates that this cmdlet lists app packages for all user accounts on the computer. To use this parameter, you must run the command by using administrator permissions.

Required	false
Position	named
Default value	False
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Name**

Description :Specifies the name of a particular package. If you specify this parameter, the cmdlet returns results for this package only. Wildcards are permitted.

Required	false
Position	1
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**PackageTypeFilter**

Description :Specifies one or more comma-separated types of packages that the cmdlet gets from the package repository. Valid values are:

Required	false
Position	named
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Publisher**

Description :Specifies the publisher of a particular package. If you specify this parameter, the cmdlet returns results only for this publisher. Wildcards are permitted.

Required	false
Position	2
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**User**

Description :Specifies a user. If you specify this parameter, the cmdlet returns a list of app packages that are installed for only the user that this cmdlet specifies. To get the list of packages for a user profile other than the profile for the current user, you must run this command by using administrator permissions. The user name can be in one of these formats:

Required	false
Position	named
Default value	None
Accept pipeline input?	True (ByValue)

Accept wildcard characters?	false

Parameter :**Volume**

Description :Specifies an AppxVolume object. If you specify this parameter, this cmdlet returns only packages that are relative to volume that this parameter specifies.

Required	false
Position	named
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Inputs

Type : System.String[]

Outputs

This cmdlet returns an AppxPackage object that contains information, including the full name of the app package.

Type : Microsoft.Windows.Appx.PackageManager.Commands.AppxPackage

Notes

Examples

Example 1: Get all app packages for every user account

This command lists the app packages that are installed for every user account on the computer.

```
PS C:\> Get-AppxPackage -AllUsers
```

Example 2: Get an app package for a specific a user

This command displays information about Package17 if it is installed in the specified user profile.

```
PS C:\> Get-AppxPackage -Name "Package17" -User
"Contoso\EvanNarvaez"
```

Cmdlet: Get-AppxPackageManifest

Synops

Gets the manifest of an app package.

Syntax

```
Get-AppxPackageManifest [-Package] <String> [[-User] <String>]
    [<CommonParameters>]
```

Description

The Get-AppxPackageManifest cmdlet gets the manifest of an app package. An app package has an .appx file name extension. The manifest is an .xml document that contains information about the package, like the package ID.

Parameters

Parameter :**Package**

Description :Specifies an AppxPackage object or the full name of a package. To get the manifest of a package on the computer that is not installed for the current user, you must run this command by using administrator permissions. Wildcards are permitted.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**User**

Description :Specifies a user. This cmdlet gets the manifest of packages that are installed for the user that this parameter specifies. To get the list of packages for a user profile other than the profile for the current user, you must run this command by using administrator permissions. The user name can be in one of these formats:

Required	false
Position	2

Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Inputs

This cmdlet accepts an array of AppxPackage objects that contain information, including the full name of the app package.

Type : Microsoft.Windows.Appx.PackageManager.Commands.AppxPackage[]

Outputs

This cmdlet returns a read-only .xml document that contains information about the app package, like the package ID.

Type : System.XML.XMLDocument

Notes

Examples

Example 1: Get the manifest for an app package

This command gets the manifest for an app package named package1_1.0.0.0_neutral__8wekyb3d8bbwe.

```
PS C:\> Get-AppxPackageManifest -Package
"package1_1.0.0.0_neutral__8wekyb3d8bbwe"
```

Example 2: Get the application ID for an app package

This command gets the application ID for an app package that has the string WinJS in the name.

```
PS C:\> (Get-AppxPackage -Name "*WinJS*" | Get-
AppxPackageManifest).package.applications.application.id
```

Example 3

This command gets the capabilities for an app package that has the string ZuneMusic in the name.

```
PS C:\> (Get-AppxPackage -Name "*ZuneMusic*" | Get-
AppxPackageManifest).Package.Capabilities
```

Cmdlet: Get-AppxVolume

Synops

Gets appx volumes for the computer.

Syntax

```
Get-AppxVolume [[-Path] <String>] -Offline [<CommonParameters>]


Get-AppxVolume [[-Path] <String>] -Online [<CommonParameters>]


Get-AppxVolume [[-Path] <String>] [<CommonParameters>]
```

Description

The Get-AppxVolume cmdlet gets a list of AppxVolume objects known to the computer. Volumes can be added by the user or a device, for instance, by using Storage Sense.

Parameters

Parameter :**Offline**

Description :Indicates that this cmdlet returns only volumes that are currently dismounted.

Required	true
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Online**

Description :Indicates that this cmdlet returns only volumes that are currently mounted.

Required	true
Position	named

Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Path**

Description :Specifies the path of the mount point of a volume. This cmdlet gets a volume at the location that this parameter specifies.

Required	false
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Outputs

Type : Microsoft.Appx.PackageManager.Commands.AppxVolume

Notes

Examples

Example 1: Get all the volumes

The command gets all the AppxVolume objects on the computer.

```
PS C:\> Get-AppxVolume
```

Example 2: Get the volume at a path

This command gets the AppxVolume at the path F:\.

```
PS C:\> Get-AppxVolume -Path F:\
```

Example 3: Get mounted volumes

This command gets only AppxVolume objects that are currently mounted on the computer.

```
PS C:\> Get-AppxVolume -Online
```

Example 4: Get volumes that are note mounted

This command gets the AppxVolume objects that not currently mounted on the computer.

```
PS C:\> Get-AppxVolume -Offline
```

Cmdlet: Invoke-CommandInDesktopPackage

Synops

{{Fill in the Synopsis}}

Syntax

```
Invoke-CommandInDesktopPackage [-PackageFamilyName] <String> [-
AppId]
    <String> [-Command] <String> [[-Args] <String>] [[-
PreventBreakaway]]
    [<CommonParameters>]
```

Description

{{Fill in the Description}}

Parameters

Parameter :**AppId**

Description :{{Fill AppId Description}}

Required	true
Position	2
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Args**

Description :{{Fill Args Description}}

Required	false
Position	4
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Command**

Description :{{Fill Command Description}}

Required	true
Position	3
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**PackageFamilyName**

Description :{{Fill PackageFamilyName Description}}

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**PreventBreakaway**

Description :{{Fill PreventBreakaway Description}}

Required	false
Position	5
Default value	False
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Inputs

System.Management.Automation.SwitchParameter

Type : System.String

Outputs

Type : System.Object

Notes

Cmdlet: Mount-AppxVolume

Synops

Mounts an appx volume.

Syntax

```
Mount-AppxVolume [-Volume] <AppxVolume[]> [-Confirm] [-WhatIf]
    [<CommonParameters>]
```

Description

The Mount-AppxVolume cmdlet mounts an AppxVolume . After you mount a volume, all apps that are deployed to that target become accessible.

Parameters

Parameter :**Volume**

Description :Specifies the AppxVolume object to mount.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Notes

Examples

Example 1: Mount a volume by using a path

This command mounts a volume at the path E:\.

```
PS C:\> Mount-AppxVolume -Volume E:\
```

Example 2: Mount a volume by using an ID

This command mounts a volume that has the specified media ID.

```
PS C:\> Mount-AppxVolume -Volume {7e62a691-398e-4fbe-819a-
64f1e407777a}
```

Cmdlet: Move-AppxPackage

Synops

Moves a package from its current location to another appx volume.

Syntax

```
Move-AppxPackage [-Package] <String[]> [-Volume] <AppxVolume> [-
Confirm]
    [-WhatIf] [<CommonParameters>]
```

Description

The Move-AppxPackage cmdlet moves a package from its current location to another AppxVolume . The new location must be a volume that Package Manager knows about and that is mounted. This cmdlet also moves your application data to the specified volume.

Parameters

Parameter :**Package**

Description :Specifies an AppxPackage object or the full name of a package. This cmdlet moves the package that this parameter specifies.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Volume**

Description :Specifies an AppxVolume object. The cmdlet moves the package to the volume that this parameter specifies.

Required	true
Position	2
Default value	None

Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Notes

Examples

Example 1: Move a package to a volume specified by a path

This command moves package that has the specified name to volume F:\. This cmdlet also moves your app data.

```
PS C:\> Move-AppxPackage -Package
"package1_1.0.0.0_neutral__8wekyb3d8bbwe" -Volume F:\
```

Example 2: Move a package to a volume specified by an ID

This command moves package that has the specified name to the volume that has the specified media ID. This cmdlet also moves your app data.

```
PS C:\> Move-AppxPackage -Package
"package1_1.0.0.0_neutral__8wekyb3d8bbwe" -Volume {d2a4d1f4-
f45a-46f3-a419-160ab52af091}
```

Cmdlet: Remove-AppxPackage

Synops

Removes an app package from a user account.

Syntax

```
Remove-AppxPackage [-Package] <String> [-AllUsers] [-Confirm] [-
WhatIf]
    [<CommonParameters>]


Remove-AppxPackage [-Package] <String> [-PreserveApplicationData]

    [-Confirm] [-WhatIf] [<CommonParameters>]


Remove-AppxPackage [-Package] <String> -User <String> [-Confirm]
[-WhatIf]
    [<CommonParameters>]
```

Description

The Remove-AppxPackage cmdlet removes an app package from a user account. An app package has an .appx file name extension.

Parameters

Parameter :**AllUsers**

Description :{{Fill AllUsers Description}}

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Package**

Description :Specifies an AppxPackage object or the full name of a package.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByValue)
Accept wildcard characters?	false

Parameter :**PreserveApplicationData**

Description :Specifies that the cmdlet preserves the application data during the package removal. The application data is available for later use.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**User**

Description :{{Fill User Description}}

Required	true
Position	named
Default value	None
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Inputs

An AppxPackage object that contain information, including the full name of the app package.

Type : System.String[]
Microsoft.Windows.Appx.PackageManager.Commands.AppxPackage

Outputs

Type : None

Notes

Cmdlet: Remove-AppxVolume

Synops

Removes an appx volume.

Syntax

```
Remove-AppxVolume [-Volume] <AppxVolume[]> [-Confirm] [-WhatIf]
    [<CommonParameters>]
```

Description

The Remove-AppxVolume cmdlet removes an AppxVolume . You can only remove a volume after there are no apps staged to it for any user. After you remove a volume, apps can no longer be added to it.

Parameters

Parameter :**Volume**

Description :Specifies the AppxVolume object to remove.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Notes

Examples

Example 1: Remove a volume by using an ID

This command removes a volume that has the specified media ID.

```
PS C:\> Remove-AppxVolume -Volume {984786d3-0cae-49de-a68f-8bedb0ca260b}
```

Example 2: Remove a volume by using a path

This command removes a volume at the path E:\.

```
PS C:\> Remove-AppxVolume -Volume E:\
```

Cmdlet: Set-AppxDefaultVolume

Synops

Specifies a default appx volume.

Syntax

```
Set-AppxDefaultVolume [-Volume] <AppxVolume> [-Confirm] [-WhatIf]

    [<CommonParameters>]
```

Description

The Set-AppxDefaultVolume cmdlet specifies a default AppxVolume . The default AppxVolume is the default target for all deployment operations on the computer. Deployment operations can specify a different non-default target volume.

Parameters

Parameter :**Volume**

Description :Specifies the path a volume. This cmdlet sets the volume that this parameter specifies to be the default deployment target for the computer.

Required	true
Position	1
Default value	None
Accept pipeline input?	True (ByPropertyName, ByValue)
Accept wildcard characters?	false

Parameter :**Confirm**

Description :Prompts you for confirmation before running the cmdlet.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Parameter :**WhatIf**

Description :Shows what would happen if the cmdlet runs. The cmdlet is not run.

Required	false
Position	named
Default value	False
Accept pipeline input?	False
Accept wildcard characters?	false

Notes

Examples

Example 1: Set a default volume by using a path

This command sets the default volume to be the volume F:\.

```
PS C:\> Set-AppxDefaultVolume -Volume F:\
```

Example 2: Set a default volume by using an ID

This command sets the default volume to be the one that has the specified media ID.

```
PS C:\> Set-AppxDefaultVolume -Volume {ef23c8d6-b13c-4c4c-
ae3b-7d5a162de9b9}
```