



PSAppDeployToolkit

Enterprise App Deployment, simplified

<https://psappdeploytoolkit.com>

v3.9.2 - May 1st 2023

Overview

- Features and Benefits
- Capability / Feature Details
- System Requirements and Support
- Donations
- Licensing

Getting Started

- Downloading & Preparing
- Folders and Files Structure
- Deploying an Application

Using the PSAppDeployToolkit

- Adding User Interface Elements
- Customizing the PSAppDeployToolkit
- Executing a Deployment Script
- Example Deployment (Basic) - Adobe Reader

Toolkit Reference - Exit Codes

Toolkit Reference - Variables

Toolkit Reference - Functions

Convert-RegistryPath

Copy-File

Disable-TerminalServerInstallMode

Enable-TerminalServerInstallMode

Execute-MSI

Execute-MSP

Execute-Process

Execute-ProcessAsUser

Exit-Script

Get-FileVersion

Get-FreeDiskSpace

Get-HardwarePlatform

Get-IniValue

Get-InstalledApplication

Get-LoggedOnUser

Get-PendingReboot

Get-RegistryKey

Get-SchedulerTask

Get-ServiceStartMode

Get-Shortcut

Get-UniversalDate

Get-UserProfiles

Get-WindowTitle

Install-MSUpdates

Install-SCCMSoftwareUpdates

Invoke-HKCURegistrySettingsForAllUsers

Invoke-RegisterOrUnregisterDLL

Invoke-SCCMTask

New-Folder

New-MsiTransform
New-Shortcut
Remove-File
Remove-Folder
Remove-InvalidFileNameChars
Remove-MSIApplications
Remove-RegistryKey
Resolve-Error
Send-Keys
Set-ActiveSetup
Set-IniValue
Set-ItemPermission
Set-PinnedApplication
Set-RegistryKey
Set-ServiceStartMode
Set-Shortcut
Show-BalloonTip
Show-DialogBox
Show-InstallationProgress
Show-InstallationPrompt
Show-InstallationRestartPrompt
Show-InstallationWelcome
Start-ServiceAndDependencies
Stop-ServiceAndDependencies
Test-Battery
Test-MSUpdates
Test-NetworkConnection
Test-PowerPoint
Test-RegistryValue
Test-ServiceExists
Update-Desktop
Update-GroupPolicy
Update-SessionEnvironmentVariables
Write-Log

Overview

The PowerShell App Deployment Toolkit (or the **PSAppDeployToolkit**) provides a framework for deploying applications in a business / corporate environment. It contains a set of well-defined functions for common application deployment tasks, as well as user interface elements for end user interaction during a deployment. It simplifies the complex scripting challenges of deploying applications in the enterprise, provides a consistent deployment experience for your end users and as a result of this, improves the overall success rate of your deployments.

Features and Benefits

Makes learning PowerShell easy

The PSAppDeployToolkit reduces the learning curve with PowerShell by providing standardized deployment template which is highly customizable, and dozens of powerful, easy to use functions to simplify common endpoint configuration tasks.

A robust and battle-tested deployment framework

The PSAppDeployToolkit was built with large enterprise environments in mind, where stability is absolutely crucial to ensure the success of mass deployments. That stability is exactly why the PSAppDeployToolkit is used by Fortune 500 companies and federal governments, banks, globally recognized brands, white-label packaging factories, defence contractors and military, in-house IT teams, consultants and managed service providers to deploy applications on millions of Windows endpoints all over the world every day.

A widely adopted standard for deployment teams

The PSAppDeployToolkit deployment template ensures applications deployments adhere to best practices and follow a standard workflow. This standardisation means you can quickly pick up work created by others on your team, and immediately understand it. It simplifies collaboration between team members and increases supportability, both in-house and from the wider community.

This is why it is recommended by, and included with [Flexera AdminStudio](#), [Master Packager](#) and [Raynet RayPack Studio](#) and why prior PSAppDeployToolkit knowledge has become a common prerequisite for endpoint engineering job postings.

Simplifies deployment troubleshooting

The PSAppDeployToolkit best practice workflow uses robust functions that handle errors and exceptions gracefully, providing extensive logging of all script actions, as well as automatically capturing Windows Installer logs in one place. This, along with optional on-screen messaging, makes it easy to quickly identify any issues in your script and quickly remediate them.

Complements and integrates with existing Endpoint Configuration Management tools

The PSAppDeployToolkit complements and integrates seamlessly with many Endpoint Configuration Management tools such as:

- System Center Configuration Manager
- Microsoft Intune / Microsoft Endpoint Manager
- Tanium Deploy
- IBM BigFix
- Ivanti Unified Endpoint Manager
- VMware Workspace ONE

Enables interaction through a User Interface which supports corporate branding

The PSAppDeployToolkit provides user interaction through customizable user interface dialogs boxes, progress dialogs and balloon tip notifications. All toolkit dialogs can be suppressed by running in Silent mode and a deployment will automatically switch to this mode if an unattended execution is detected, e.g there is currently no user logged into the computer, or the deployment is being run from an Operating System Deployment (OSD) Task Sequence.

Enhances endpoint security posture

Leveraging the PSAppDeployToolkit's custom branding capabilities enhances your deployments with a consistent and brand compliant look and feel, as well as allowing you to easily integrate your corporate nomenclature with the user interface text. Nefarious installers and malware attempting to mimic well known application installers, are more

likely to be noticed as a red flag by end users. Incorporating security messaging to report suspicious activity to your Service Desk, can highlight attempted breaches that might not be picked up by commodity Antivirus or Endpoint Detection and Response (EDR) tools.

Localizable

The UI is localized in more than 20 languages and more can easily be added using the XML configuration file.

Updateable

The logic engine and functions are separated from per-application scripts, so that you can update the PSAppDeployToolkit when a new version is released and maintain backwards compatibility with your deployment scripts.

Extensible

The PSAppDeployToolkit can be easily extended to add custom scripts and functions.

Helpful

The PSAppDeployToolkit provides detailed logging of all actions performed and even includes a graphical console to browse the help documentation for the PSAppDeployToolkit functions.

Capability / Feature Details

Let's take a look at some of the capabilities that get exposed automatically when you use the PSAppDeployToolkit.

Pre-Built PowerShell Functions

- Pre-built functions come with automated logging - so you can quickly find problems in your scripts if they occur.
- Provides the ability to execute any type of setup (Windows Installer or Executable-based) with automated exit code handling, as well as MSI-based installers having their logs captured and stored alongside the PSAppDeployToolkit logs.
- Mass remove MSI applications with a partial match (e.g. remove all versions of all MSI applications which match "Office")

- Perform MEMCM actions such as Machine and User Policy Refresh, Inventory Update and Software Update
- Supports installation of applications on Citrix / Remote Desktop Session Host Servers
- Update Group Policy
- Copy / Delete Files
- Get / Set / Remove Registry Keys and Values
- Get / Set INI File Keys and Values
- Check File versions
- Pin or Unpin applications to the Start Menu or Task Bar
- Create Start Menu Shortcuts
- Register / Unregister DLL files
- Refresh desktop icons / environment variables
- Test network connectivity
- Test power connectivity
- Check whether a PowerPoint slide show is running in full screen presentation mode

User Experience

- An interface to prompt the user to close specified applications that are open prior to starting the application deployment. The user is prompted to save their documents and has the option to close the programs themselves, have the PSAppDeployToolkit close the programs, or optionally defer. Optionally, a countdown can be displayed until the applications are automatically closed.
- The ability to allow the user to defer an installation X number of times, X number of days or until a deadline date is reached.
- The ability to prevent the user from launching the applications that need to be closed while the application installation is in progress.
- An indeterminate progress dialog with customizable message text that can be updated throughout the deployment.
- A restart prompt with an option to restart later or restart now and a countdown to automatic restart.
- The ability to notify the user if disk space requirements are not met.
- Custom dialog boxes with options to customize title, text, buttons & icon.

- Balloon tip notifications to indicate the beginning and end of an installation and the success or failure of an installation.
- Branding of the above UI components using a custom logo icon and banner for your own Organization.
- The ability to run in interactive, silent (no dialogs) or non-interactive mode (default for running MEMCM task sequence or session 0).
- The UI is localized into several languages and more can be easily added using the XML configuration file.

3rd Party Integration

- Handles MEMCM exit codes, including time sensitive dialogs supporting "MEMCM Fast Retry" - providing more accurate Reporting (no more Failed due to timeout errors).
- Ability to prevent reboot codes (3010) from being passed back to MEMCM, which would cause a reboot prompt.
- Supports the MEMCM application model by providing an install and uninstall deployment type for every deployment script.
- Bundle multiple application installations to overcome the supported limit of 5 applications in the MEMCM application dependency chain.
- Compared to compiled deployment packages, e.g. WiseScript, the PSAppDeployToolkit utilizes the MEMCM cache correctly and MEMCM Distribution Point bandwidth more efficiently by using loose files.

Documentation

- A graphical console to easily browse the PSAppDeployToolkit Cmdlet documentation.
 - Further documentation available on GitHub Wiki.
 - Complete documentation included as PDF for use on e-readers / offline.
-

System Requirements and Support

The PSAppDeployToolkit 3.x is primarily developed on the latest Windows Client operating system (currently Windows 11). However, enterprise device lifecycles may be driven by external factors, sometimes requiring long-term service of older operating system version. As such we put each release through testing on a wide range of Operating

Systems from Windows XP to Windows 11 (and their Windows Server equivalents) in order to provide enterprise-wide compatibility.

The system requirements are as follows:

- PowerShell 2.0
- Windows NT 5.1 and above

While we have attempted to maintain this backwards compatibility through the life cycle of the PSAppDeployToolkit, the degree of testing performed across older Operating Systems such as XP and Vista is limited as the bulk of testing is performed on the latest OS versions. However, the PSAppDeployToolkit has widespread adoption in the enterprise from SMEs to large multinationals so there is safety in numbers and the assurance that the PSAppDeployToolkit has been put through its paces VERY extensively on a large number of Windows clients around the globe.

From the customers that we are aware of using the PSAppDeployToolkit, we estimate it is used to deploy software on over 5 million endpoints.

Donations

We have invested our spare time in the creation and ongoing development, maintenance and support of this community tool on a voluntary basis - it is not part of our day jobs. Donations to the project are welcome, please visit the following page for details on making a contribution:

<http://psappdeploytoolkit.com/donate>

Licensing

PSAppDeployToolkit - Copyright ©, 2023 The PSAppDeployToolkit Team.

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this program. If not, see <http://www.gnu.org/licenses>.

Getting Started

Downloading & Preparing

The PSAppDeployToolkit doesn't actually need to be installed, but we did need an 'Installation' section for our documentation :) The easiest way to get the PSAppDeployToolkit is to [download the latest version from GitHub](#).

The PSAppDeployToolkit package is provided in a Zip file archive. Once extracted, you'll see a folder structure similar to that outlined in the next section.

Folders and Files Structure

The PSAppDeployToolkit package template is made up of the following folders and files, and should reflect what you see when you unzip the PSAppDeployToolkit archive:

Root Folder

Folder Name	Description
Toolkit/	Contains the PSAppDeployToolkit itself and all of the files / folder structure required to create a deployment package.
Examples/	Contains example deployments for both Adobe Reader and Microsoft Office 2013.
PSAppDeployToolkit.pdf	The PSAppDeployToolkit documentation in PDF format.
CHANGELOG.txt	The changelog for the current release of the PSAppDeployToolkit.
LICENSE.txt	The LGPL license under which the PSAppDeployToolkit is distributed.

Toolkit Folder

This contains all of the files needed to create a new deployment package

Folder / File Name	Description
AppDeployToolkit/	Contains the PSAppDeployToolkit dependency files.
Files/	Contains the primary installation file(s), e.g. MSI file. This folder is empty by default.
SupportFiles/	Contains any supporting resources or assets, e.g. files you need to copy to the target machine using the PSAppDeployToolkit during deployment. This folder is empty by default.
Deploy_Application.ps1	PowerShell script that contains the logic to perform the actual install / uninstall / repair. This is the only file that needs to be modified, depending on your level of customization.
Deploy-Application.exe	An optional executable that can be used to launch the Deploy-Application.ps1 script without opening a PowerShell console window. Supports passing command-line parameters to the script.
Deploy-Application.exe.config	A .NET configuration file required for Deploy-Application.exe to function correctly.

AppDeployToolkit Folder

File Name	Description
-----------	-------------

File Name	Description
AppDeployToolkitBanner.png	Contains the default PSAppDeployToolkit branded header. To brand the PSAppDeployToolkit User Interface with your own custom / corporate banner, replace this file with your own image. The file must be in PNG format and must be 450 x 50 in size.
AppDeployToolkitConfig.xml	Contains configurable options referenced by the AppDeployToolkitMain.ps1 script, such as MSI switches and User Interface messages, which are customizable and localized in several languages. This is intended to be a static file that is configured once, not on a per-application basis.
AppDeployToolkitExtensions.ps1	This is an optional PowerShell script that can be used to extend the PSAppDeployToolkit functionality with custom functions. It is automatically dot-sourced by the AppDeployToolkitMain.ps1 script.
AppDeployToolkitHelp.ps1	This is a script that displays a help console to browse the functions included in the PSAppDeployToolkit and copy and paste examples in to your deployment script.
AppDeployToolkitLogo.ico	Contains the default PSAppDeployToolkit branded icon. To brand the balloon notifications and UI window title bars with your own custom / corporate logo, replace this file with your own icon.
AppDeployToolkitMain.cs	Contains additional classes and methods referred by the AppDeployToolkitMain.ps1 script.
AppDeployToolkitMain.ps1	Contains all of the functions and logic used by the installation script. By Separating the logic from the installation script, we can obfuscate away the complex code and make enhancements independently of the installation scripts that contain per-application actions.

Deploying an Application

Deployment Phases

The `Deploy-Application.ps1` script is the only one you need to modify to deploy your application.

The script is broken down into the following sections:

- **Initialization** - Variables such as App Vendor, App Name, App Version
- **Pre-Installation** - Close applications, uninstall or clean-up previous versions
- **Installation** - Install the primary application, or components of the application
- **Post-Installation** - Drop additional files, registry tweaks
- **Uninstallation** - Uninstall/rollback the changes performed in the install section.
- **Repair** - Repair the changes performed in the install section.

Zero-Configuration MSI Deployment

The PSAppDeployToolkit has a zero-config MSI install feature which allows you to quickly execute an installation with zero configuration of the `Deploy-Application.ps1` file. To use this feature:

1. Place your MSI file into the `Files` folder of the PSAppDeployToolkit. This method only support the installation of one MSI, so if more than one MSI is found, then only the first one is selected.
 2. If you have an MST file, then place it into the `Files` folder of the PSAppDeployToolkit. The MST file must have the same name as the MSI file. For example, if your MSI file name is `test01.msi`, then the MST file must be named `test01.mst`.
 3. If you have any MSP files, then place it into the `Files` folder of the PSAppDeployToolkit. You can place more than one MSP file in the folder, but you must name the files in alphabetical order to control the order in which they are installed. MSP file will be installed in alphabetical order.
-

Using the PSAppDeployToolkit

Adding User Interface Elements

The user interface consists of several components detailed below. It can be branded with a custom logo and banner, and all on-screen messaging can be customized in `AppDeployToolkitConfig.xml`. Apart from the default of English, the User Interface has also been localized in the following languages:

- Danish
- Dutch
- French
- German
- Italian
- Japanese
- Norwegian
- Portuguese
- Spanish
- Swedish

Additional languages can be easily added in the XML configuration file.

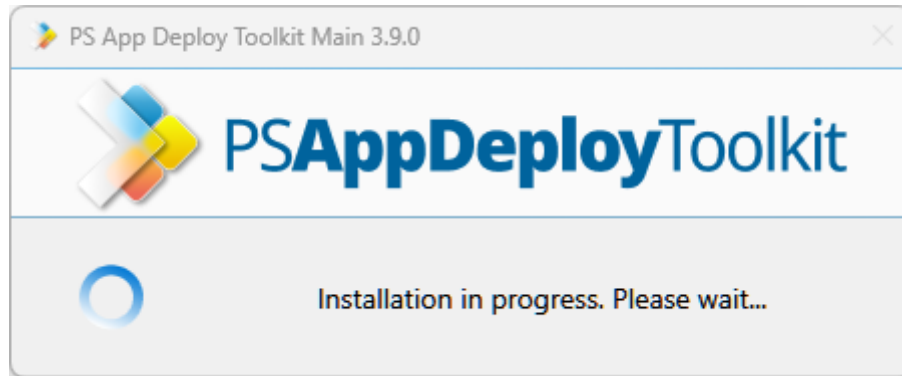
The language used by the PSAppDeployToolkit User Interface is selected automatically based on the language culture of the operating system, so the same `AppDeployToolkitConfig.xml` file can be used in a multi-language environment.

The user interface can be suppressed by specifying the deploy mode parameter as follows:

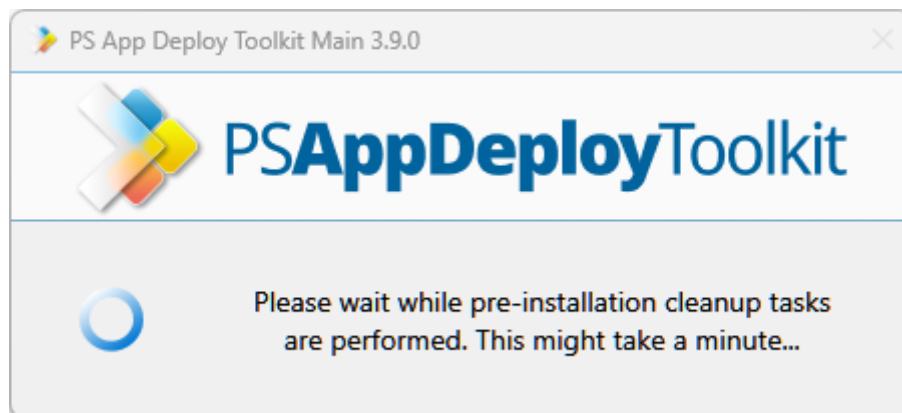
```
Deploy-Application.ps1 -DeployMode "Silent"
```

Installation Progress

The installation progress message displays an indeterminate progress ring to indicate an installation is in progress and display status messages to the end user. This is invoked using the Show-InstallationProgress Function.

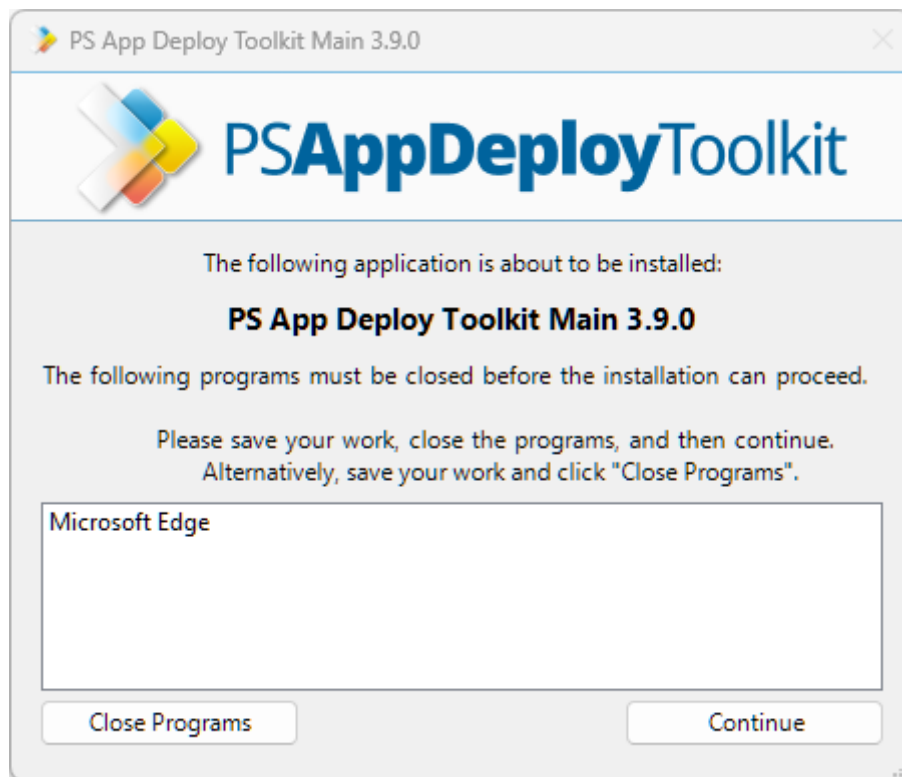


The progress message can be dynamically updated to show the installation stage or to display custom messages, using Show-InstallationProgress.

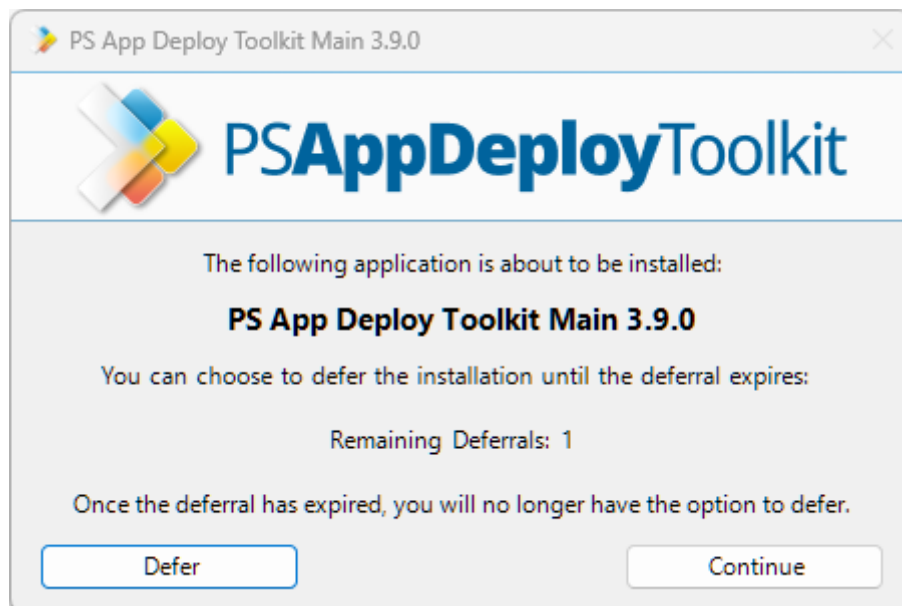


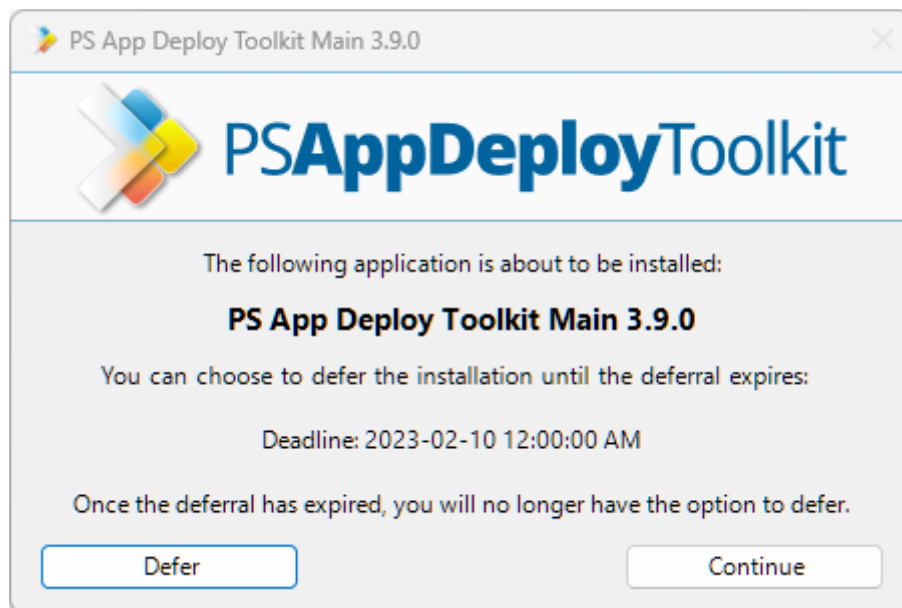
Installation Welcome Prompt

The application welcome prompt can be used to display applications that need to be closed, an option to defer and a countdown to closing applications automatically. Use the Show-InstallationWelcome function to display the prompts shown below.

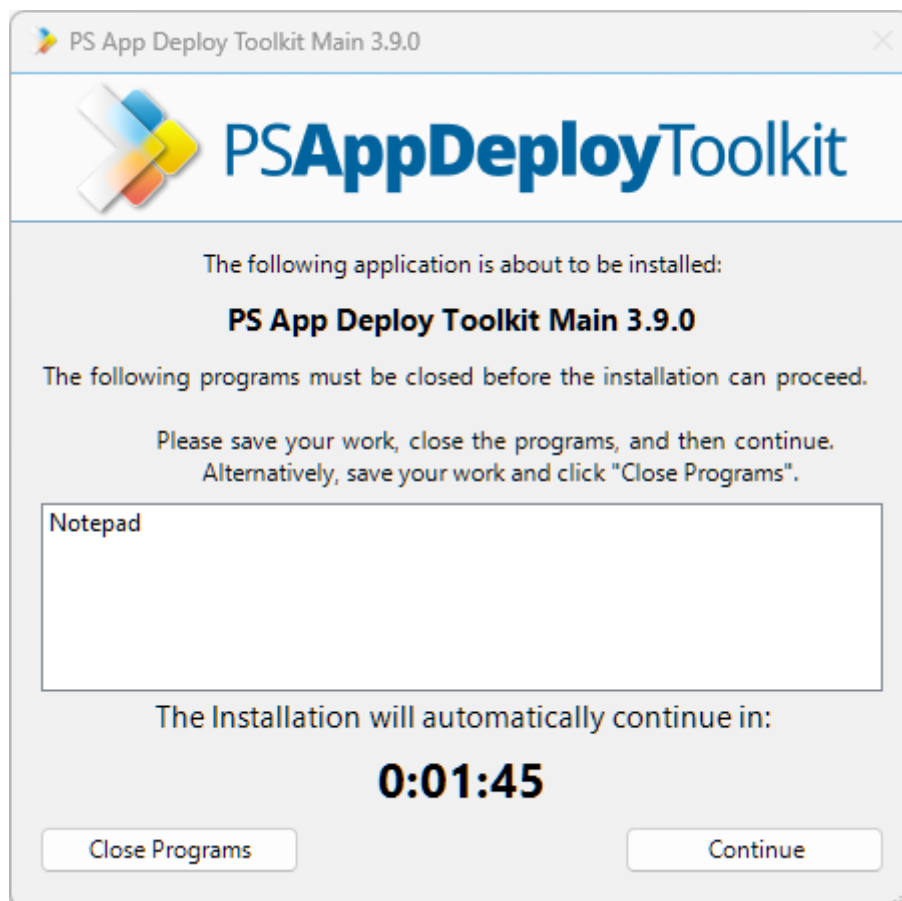


Welcome prompt with close programs option and defer option:





Welcome prompt with close programs options and countdown to automatic closing of applications:

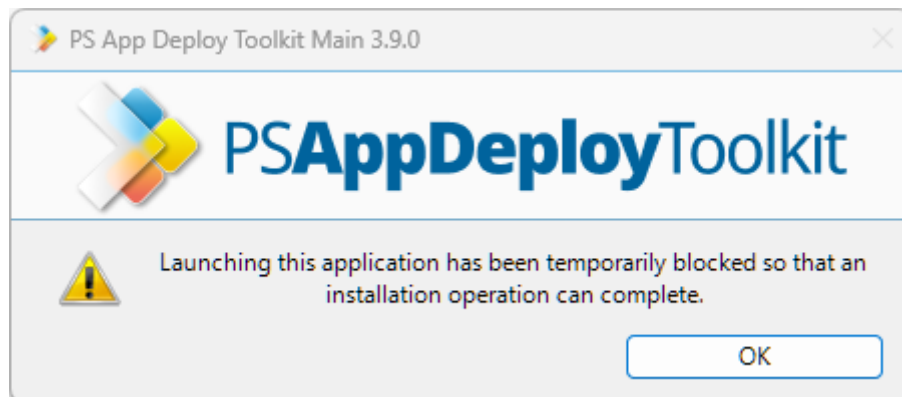


Welcome prompt with just a defer option:



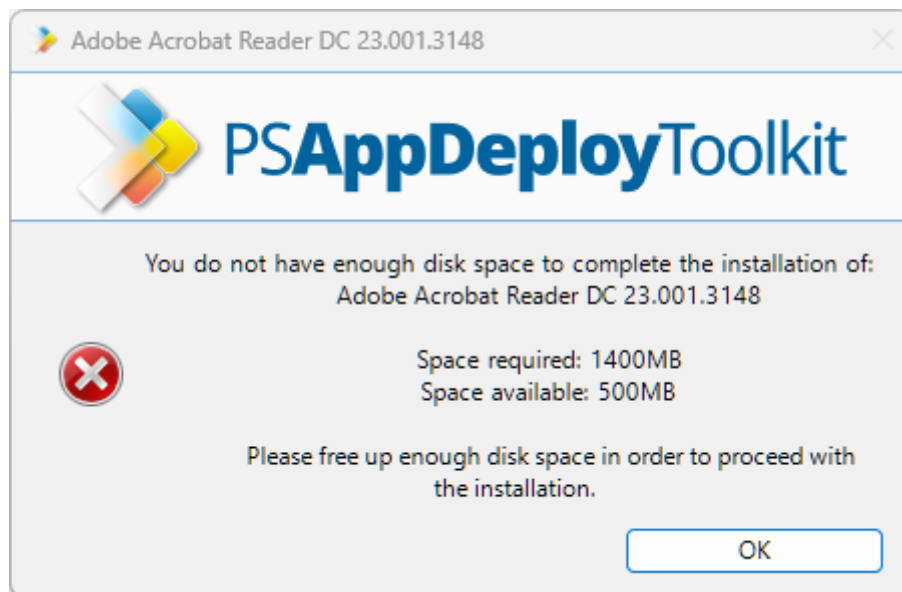
Block Application Execution

If the block execution option is enabled (see `Show-InstallationWelcome` function), the user will be prompted that they cannot launch the specified application(s) while the installation is in progress. The application will be unblocked again once the installation has completed.



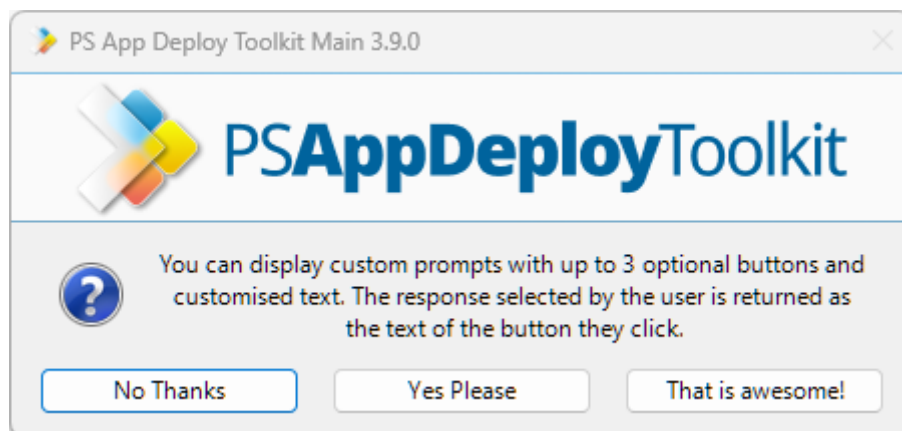
Disk Space Requirements

If the `CheckDiskSpace` parameter is used with the `Show-InstallationWelcome` function and the disk space requirements are not met, the following prompt will be displayed and the installation will not proceed.

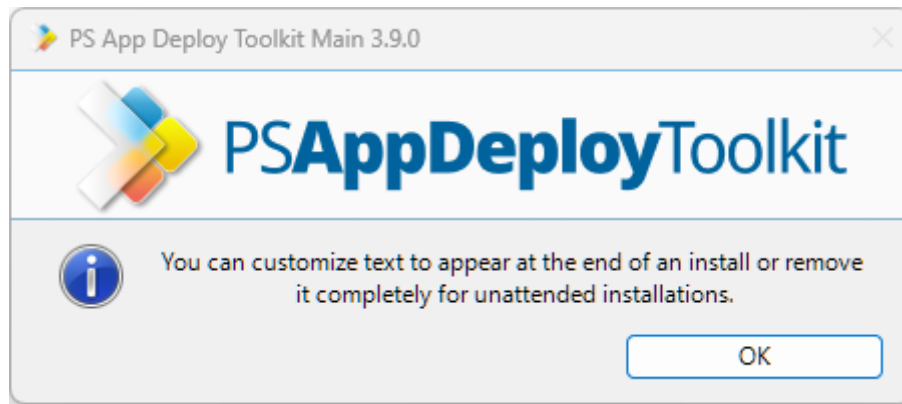


Custom Installation Prompt

A custom prompt with the PSAppDeployToolkit branding can be used to display messages and interact with the user using the Show-InstallationPrompt function. The title and text is customizable and up to 3 customizable buttons can be included on the prompt as well as optional system icons, e.g.

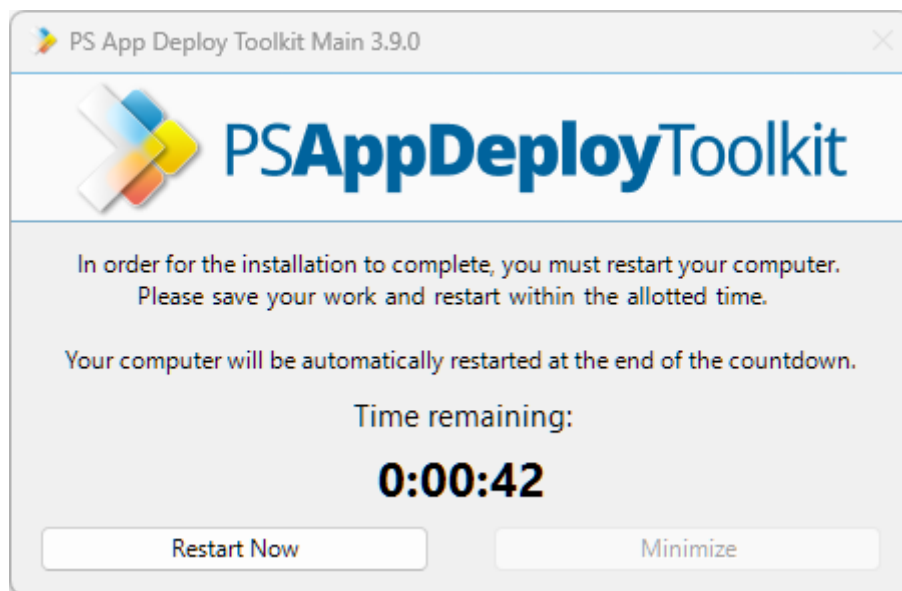


Additionally, the prompt can be displayed asynchronously, e.g. to display a message at the end of the installation but allow the installation to return the exit code to the parent process without waiting for the user to respond to the message.



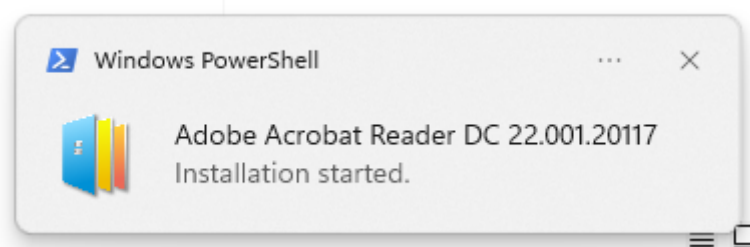
Installation Restart Prompt

A restart prompt can be displayed with a countdown to automatic restart using the `Show-InstallationRestartPrompt`. Since the restart prompt is executed in a separate PowerShell session, the PSAppDeployToolkit will still return the appropriate exit code to the parent process.



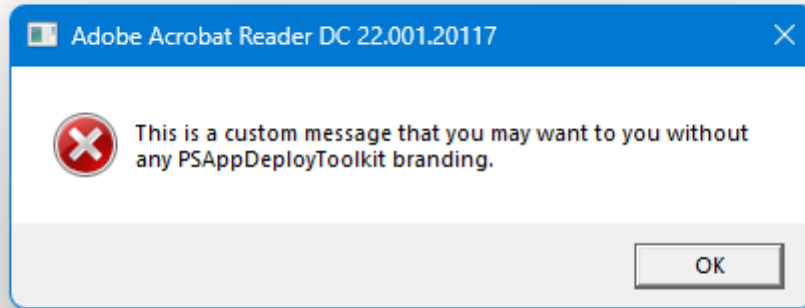
Balloon Tip Notifications

Balloon tip notifications are displayed in the system tray automatically at the beginning and end of the installation. These can be turned off in the XML configuration.



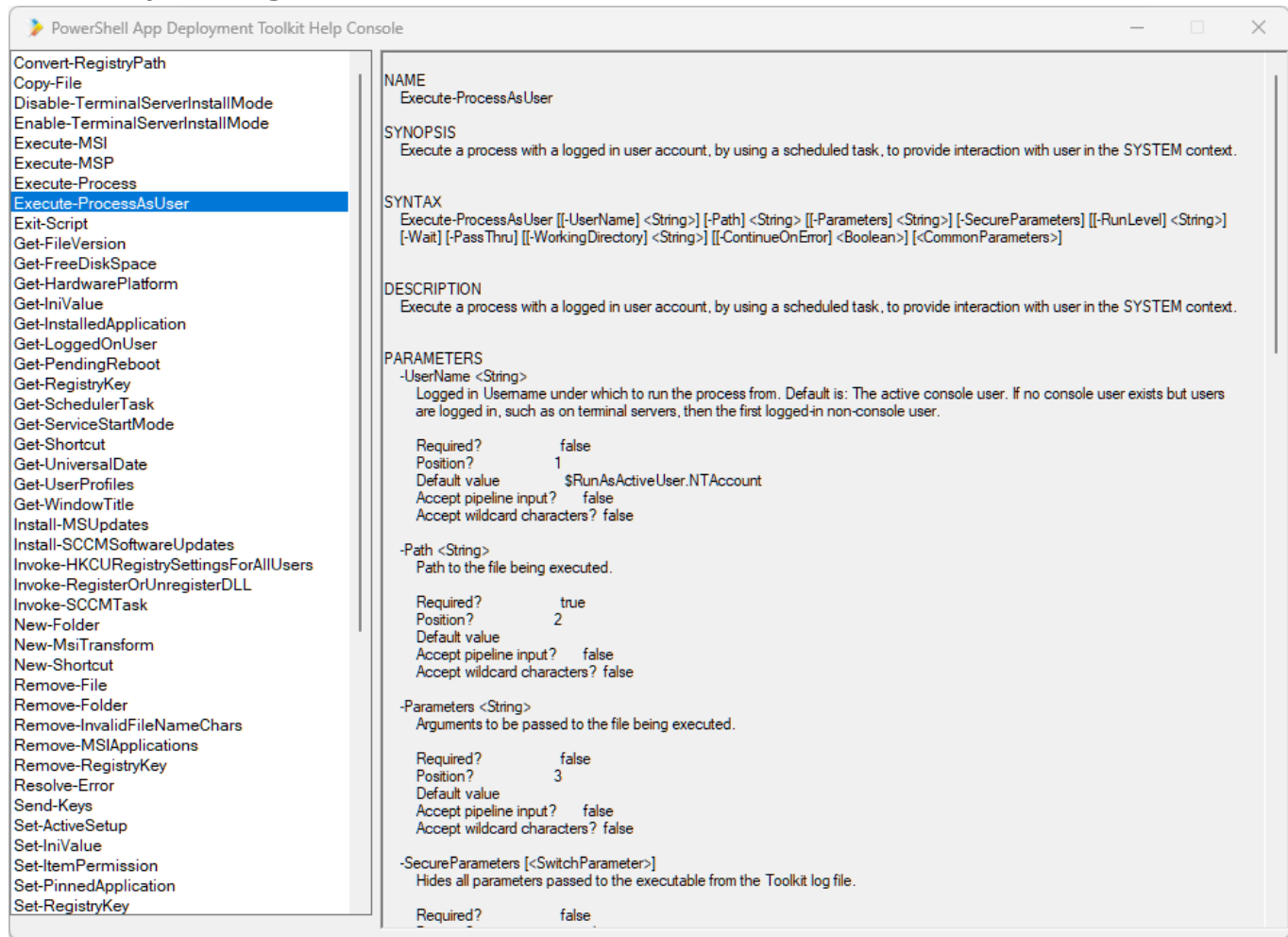
Custom Dialog box

A generic dialog box to display custom messages to the user without the PSAppDeployToolkit branding using the function `Show-DialogBox`. This can be customized with different system icons and buttons.



Toolkit Help Console

The PowerShell App Deployment Toolkit Help Console. This is a separate script and can be invoked by running AppDeployToolkitHelp.ps1



Customizing the PSAppDeployToolkit

Aside from customizing the Deploy-Application.ps1 script to deploy your application, no configuration is necessary out of the box. The following components can be configured as required:

AppDeployToolkitConfig.xml

Configure the default UI messages, MSI parameters, log file location, whether Admin rights should be required, whether log files should be compressed, log style (CMTrace or Legacy), max log size, whether debug messages should be logged, whether log entries should be written to the console, whether toolkit should re-launch as elevated logged-on console user when in SYSTEM context, whether toolkit should fall back to SYSTEM context if failure to launch toolkit as user, and whether toolkit should attempt to launch as a non-

console logged on user (e.g. user logged on via terminal services) when in SYSTEM context.

AppDeployToolkitLogo.ico

To brand the balloon notifications and UI window title bars with your own custom/corporate logo, replace the AppDeployToolkitLogo.ico file with your own .ico file (retaining the file name)

AppDeployToolkitBanner.png

To brand the PSAppDeployToolkit UI prompts with your own custom/corporate banner, replace the AppDeployToolkitBanner.png file with your own .png file (retaining the file name). The file must be in PNG format and must be 450 x 50 in size.

Logging

The PSAppDeployToolkit generates extensive logging for all toolkit and MSI operations.

The default log folder for the PSAppDeployToolkit and MSI log files can be specified in the XML configuration file. The default folder is C:\Windows\Logs\Software.

The PSAppDeployToolkit log file is named after the application with _PSAppDeployToolkit appended to the end, e.g.

- Oracle_JavaRuntime_1.7.0.17_EN_01_PSAppDeployToolkit.log

All MSI actions are logged and the log file is named according to the MSI file used on the command line, with the action appended to the log file name. For uninstallations, the MSI product code is resolved to the MSI application name and version to keep the same log file format, e.g.

- Oracle_JavaRuntimeEnvironmentx86_1.7.0.17_EN_01_Install.log
- Oracle_JavaRuntimeEnvironmentx86_1.7.0.17_EN_01_Repair.log
- Oracle_JavaRuntimeEnvironmentx86_1.7.0.17_EN_01_Patch.log
- Oracle_JavaRuntimeEnvironmentx86_1.7.0.17_EN_01_Uninstall.log

Log Compression

One of the PSAppDeployToolkit Options in the AppDeployToolkitConfig.xml file is CompressLogs. Enabling this option will create a temporary logging folder where you can save all of the log files you want to include in the single ZIP file that will be created from this folder.

To enable the CompressLogs feature, set the follow option in AppDeployToolkitConfig.xml to True:

```
<Toolkit_CompressLogs>True</Toolkit_CompressLogs>
```

When set to True, the following happens:

- Both toolkit and MSI logs are temporally placed in \$envTemp\$installName which gets cleaned up at the end of the install.
- At the end of the install / uninstall, the logs are compressed into a new zip file which is placed in the LogFolder location in the config file.
- The Zip file name indicates whether it is an Install / Uninstall and has the timestamp in the filename so previous logs do not get overwritten.
- If your package creates other log files, you can send them to the temporary logging folder at \$envTemp\$installName.

Executing a Deployment Script

There are two ways to launch the PSAppDeployToolkit for deployment of applications.

- Launch Deploy-Application.ps1 PowerShell script as administrator.
- Launch Deploy-Application.exe as administrator. This will launch the Deploy-Application.ps1 PowerShell script without opening a PowerShell command window.
Note, if the x86 PowerShell is required (for example, if CAPICOM or another x86 library is needed), launch Deploy-Application.exe /32.

Deploy an application for installation


```
Deploy-Application.ps1
```

Deploy an application for uninstallation in silent mode

```
Deploy-Application.ps1 -DeploymentType 'Uninstall' -DeployMode 'Silent'
```

Deploy an application for uninstallation using PowerShell x86, suppressing the PowerShell console window and deploying in silent mode.

```
Deploy-Application.exe /32 -DeploymentType 'Uninstall' -DeployMode 'Silent'
```

Deploy an application for installation, suppressing the PowerShell console window and allowing reboot codes to be returned to the parent process.

```
Deploy-Application.exe -AllowRebootPassThru
```

Deploy an application with a custom name instead of Deploy-Application.ps1.

```
Deploy-Application.exe 'Custom-Script.ps1'
```

Remove an application with a custom name and custom location for the script file.

```
Deploy-Application.exe -Command 'C:\Testing\Custom-Script.ps1' -DeploymentType 'Uninstall'
```

Toolkit Parameters

The following parameters are accepted by `Deploy-Application.ps1`:

-DeploymentType

Specify whether to install or uninstall the application.

```
-DeploymentType 'Install' ## default
```

```
-DeploymentType 'Uninstall'
```

-DeployMode

Specify the installation will be run in Interactive, Silent or NonInteractive mode:

- Interactive = Shows dialogs
- NonInteractive = Very silent, i.e. no blocking apps. This is automatically set if it is detected that the process is not running in the user session and it is not possible for anyone to provide input using a mouse or keyboard.
- Silent = No dialogs (progress and balloon tip notifications are suppressed)

```
-DeployMode 'Interactive' ## default
```

```
-DeployMode 'Silent'
```

```
-DeployMode 'NonInteractive'
```

-AllowRebootPassthru

Specify whether to allow the 3010 exit code (reboot required) to be passed back to the parent process (e.g. MEMCM) if detected during an installation. If a 3010 code is passed to MEMCM, the MEMCM client will display a reboot prompt. If set to false, the 3010 return code will be replaced by a "0" (successful, no restart required).

```
-AllowRebootPassThru $true ## default
```

```
-AllowRebootPassThru $false
```

-TerminalServerMode

Changes to user install mode and back to user execute mode for installing/uninstalling applications on Remote Desktop Session Host/Citrix servers

```
-TerminalServerMode $true ## default
```

```
-TerminalServerMode $false
```

-DisableLogging

Disables logging to file for the script.

```
-DisableLogging
```

This is a switch parameter, so setting this enables it. When not present, the default is false.

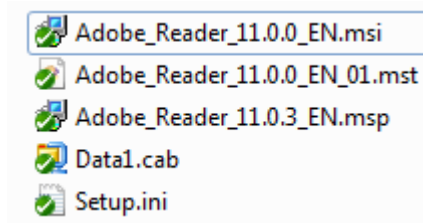
Example Deployment (Basic) - Adobe Reader

In this example, we will build an Adobe Reader installation which provides the following benefits over using a standard MSI based MEMCM deployment:

- The ability to defer the installation up to 3 times
- The ability to close any applications that could cause errors during the installation
- Verification that the required disk space is available
- Full removal of any previous version of Adobe Reader (to prevent issues sometimes seen when doing an MSI upgrade, i.e. Missing previous installation source files)
- Installation of any subsequent patches required after the base MSI installation

This example is provided as a script with the PSAppDeployToolkit, in the `Examples` folder.

- Copy the application source files in to the `Files` folder, e.g.



Script Customisation

Customize the `Deploy-Application.ps1` script using the example code below:

Initialization

Populate these variables with the application and script details:

```
$appVendor = 'Adobe'
$appName = 'Reader'
$appVersion = '11.0.3'
$appArch = ''
$appLang = 'EN'
$appRevision = '01'
$appScriptVersion = '1.0.0'
$appScriptDate = '08/07/2013'
$appScriptAuthor = 'Your Name'
```

Pre-Install

Copy the following into the Pre-Install section:

```
## Prompt the user to close the following applications if they are running and
## allow the option to defer the installation up to 3 times:
Show-InstallationWelcome -CloseApps 'iexplore,AcroRd32,cidaemon' -AllowDefer -
DeferTimes 3

## Show Progress Message (with the default message)
Show-InstallationProgress

## Remove any previous versions of Adobe Reader
Remove-MSIApplications -Name 'Adobe Reader'
```

Installation

Copy the following into the Installation section:

```
## Install the base MSI and apply a transform
Execute-MSI -Action Install -Path 'Adobe_Reader_11.0.0_EN.msi' -Transform
'Adobe_Reader_11.0.0_EN_01.mst'

## Install the patch
Execute-MSI -Action Patch -Path 'Adobe_Reader_11.0.3_EN.msp'
```

Post-Installation

Copy the following into the Post-Install section:

```
## No actions required here
```

Uninstallation

Copy the following into the Uninstallation section:

```
## Prompt the user to close the following applications if they are running:
Show-InstallationWelcome -CloseApps 'iexplore,AcroRd32,cidaemon'

## Show Progress Message (with a message to indicate the application is being
uninstalled)
Show-InstallationProgress -StatusMessage "Uninstalling Application $installTitle.
Please Wait..."

## Remove this version of Adobe Reader
Execute-MSI -Action Uninstall -Path '{AC76BA86-7AD7-1033-7B44-AB0000000001}'
```

Testing the Deployment Script

- Install the application by running:

```
Deploy-Application.exe
```

- Uninstall the application by running:

```
Deploy-Application.exe -DeploymentType "Uninstall"
```

Deploying through MEMCM using the Package model

- Copy the installation files to a network location accessible by MEMCM.
- Create a new Package:

New Package Wizard

General

General

Data Source

Data Access

Distribution Settings

Reporting

Security

Summary

Progress

Confirmation

Adobe Reader 11.0.3 English

Name: Reader

Version: 11.0.3

Manufacturer: Adobe

Language: English

Comment:

< Previous Next > Finish Cancel

- Set the package source folder accordingly:

New Package Wizard

Data Source

General
Data Source
Data Access
Distribution Settings
Reporting
Security
Summary
Progress
Confirmation

Specify whether this package contains source files. If it does, specify the initial location of the files and set additional source file options.

☒ This package contains source files

Source version:

Source directory
\\R8GKE6E\c\$\Adobe_Reader_11.0.3_EN_01 Set...

☐ Use a compressed copy of the source directory
☒ Always obtain files from the source directory

☐ Update distribution points on a schedule Schedule...

☐ Persist content in the client cache
☐ Enable binary differential replication

< Previous Next > Finish Cancel

- Accept the defaults for the rest of the package (or modify according to your environment)
- Distribute the content of the package to the relevant Distribution Points
- Create a new Program for the package:

New Program Wizard

General

General

Requirements

Environment

Advanced

Windows Installer

MOM Maintenance

Summary

Progress

Confirmation

Name: Adobe Reader 11.0.3 EN 01 PSAppDeployToolkit

Your use of software deployed by ConfigMgr may be subject to license terms. You should review any applicable license terms prior to deploying software.

Comment:

Command line: Deploy-Application.exe Browse...

Start in:

Run: Normal

After running: No action required

Category:

< Previous Next > Finish Cancel

- Accept the defaults for the requirements of the program (or modify according to your environment)
- On the Environment page, ensure you use a combination of settings that allows the user to interact with the application. Failure to do so will result in the application installing silently:

New Program Wizard

Environment

General
Requirements
Environment
Advanced
Windows Installer
MOM Maintenance
Summary
Progress
Confirmation

A program may require certain conditions to be true before it can run. Specify the conditions that must be met for the program to run.

Program can run: Only when a user is logged on

Run mode

☐ Run with user's rights

☒ Run with administrative rights

☒ Allow users to interact with this program

Drive mode

☒ Runs with UNC name

☐ Requires drive letter

☐ Requires specific drive letter (example: Z):

☐ Reconnect to distribution point at login

< Previous Next > Finish Cancel

- Accept the defaults for the rest of the program (or modify according to your environment)
- Create a new Advertisement for the Package and set your target collection accordingly:

New Advertisement Wizard

General

General

Schedule

Distribution Points

Interaction

Security

Summary

Progress

Confirmation

Name: Adobe Reader 11.0.3 EN 01 Deploy With 3 Deferrals

Comment:

Package: Adobe Reader 11.0.3 English Browse...

Program: Adobe Reader 11.0.3 EN 01 PSAppDeploy1


Collection: Dan Testing Browse...

☒ Include members of subcollections

< Previous Next > Finish Cancel

- Set a recurring schedule for the Mandatory Assignment. This dictates how frequently the application should attempt to install. Additionally, ensure that "Rerun if failed previous attempt" is enabled. These settings are required when using the deferral system and ensure that if a user defers the install, the install will retry after the specified interval:

New Advertisement Wizard ✕



Schedule

General

Schedule

Distribution Points

Interaction

Security

Summary

Progress

Confirmation

Specify when the program will be advertised to members of the target collection. You can also create an assignment to make the program mandatory.

Advertisement start time:

2013-10-10

📅

9:01 AM

⬆️⬇️⬆️

☐ UTC

☐ Advertisement expires:

2014-04-10

📅

9:01 AM

⬆️⬇️⬆️

☐ UTC

Mandatory assignments: ☀️ 📅 ✕

Occurs every 6 hour(s) effective 2013-10-10 9:02 AM

☐ Enable Wake On LAN

☐ Ignore maintenance windows when running program

☐ Allow system restart outside maintenance windows


Priority: Medium ▼

Program rerun behavior: Rerun if failed previous attempt ▼

< Previous
Next >
Finish
Cancel

- When prompted with the following dialog box, select Yes:

Advertisement ✕



Because you have specified more than one mandatory assignments for this advertisement, we recommend setting the program rerun behavior to 'Always rerun program'.

Do you want to keep your current setting?

To keep your current setting, click Yes.

To set program rerun behavior to 'Always rerun program', click No.

To return to the advertisement wizard, click Cancel.

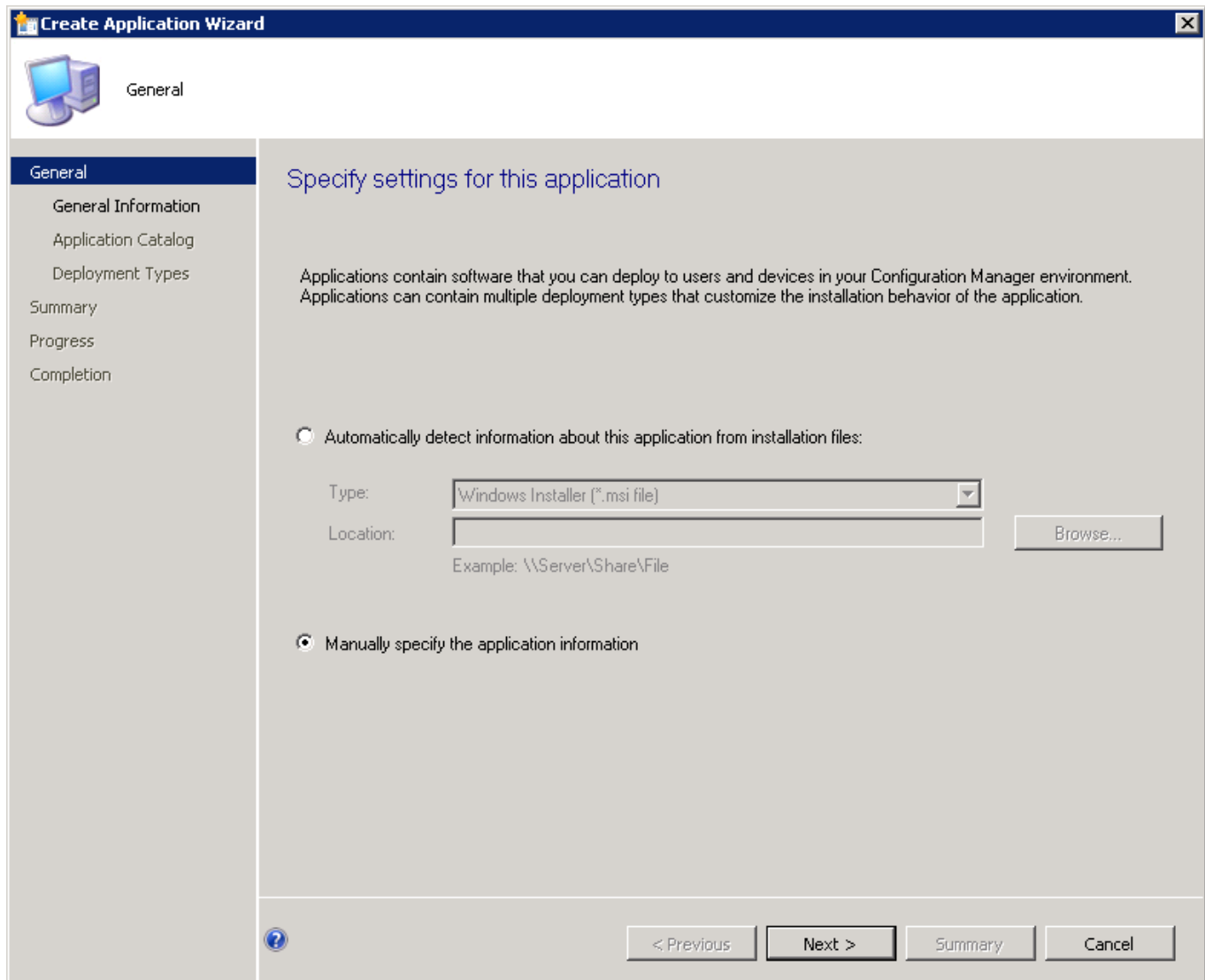
Yes
No
Cancel

- Accept the defaults for the rest of the advertisement (or modify according to your environment). The deployment should start on your target machines shortly.

Deploying through MEMCM using the Application model

Important Note: The MEMCM Application Model does not have the flexibility to schedule Mandatory Assignments on a recurring schedule like MEMCM packages do. Instead, this is determined by the frequency of Software Deployment evaluation cycle in the MEMCM Agent Custom Settings. You can modify this to reduce the time from the default of once a day, however this may increase the load on your MEMCM servers and clients, and is not configurable on a per application basis.

- Copy the installation files to a network location accessible by MEMCM.
- Create a new Application and manually specify the application information:



The screenshot shows the 'Create Application Wizard' dialog box with the 'General' tab selected. The left sidebar contains a tree view with the following items: General (selected), General Information, Application Catalog, Deployment Types, Summary, Progress, and Completion. The main area is titled 'Specify settings for this application' and contains the following text: 'Applications contain software that you can deploy to users and devices in your Configuration Manager environment. Applications can contain multiple deployment types that customize the installation behavior of the application.'

There are two radio buttons for selecting the method to specify application information:

- ☐ Automatically detect information about this application from installation files:
- ☒ Manually specify the application information

Under the 'Automatically detect...' option, there are two fields:

- Type: A dropdown menu showing 'Windows Installer (*.msi file)'.
- Location: A text box with an example path '\\Server\Share\File' and a 'Browse...' button.

At the bottom of the dialog, there are four buttons: '< Previous', 'Next >', 'Summary', and 'Cancel'.

- Populate the application details accordingly:

Create Application Wizard

General Information

General Information

Application Catalog

Deployment Types

Summary

Progress

Completion

Specify information about this application

Name: Adobe Reader 11.0.3

Administrator comments:

Manufacturer: Adobe Software version: 11.0.3 EN 01

Optional reference:

Administrative categories: "Utilities" Select...

☐ Date published: 10/10/2013

☐ Allow this application to be installed from the Install Application task sequence action without being deployed

Specify the administrative users who are responsible for this application.

Owners: dcunningha002 Browse...

Support contacts: dcunningha002 Browse...

< Previous Next > Summary Cancel

- Populate the application catalog details if required
- Add a new Deployment Type and manually specify the deployment type information:

Create Deployment Type Wizard

General

General

General Information
Content
Detection Method
User Experience
Requirements
Dependencies
Summary
Progress
Completion

Specify settings for this deployment type


Deployment types include information about the installation method and source files for this application.

Type:

☐ Automatically identify information about this deployment type from installation files

Location:
Example: \\Server\Share\File

☒ Manually specify the deployment type information



- Populate the deployment type details accordingly:

The screenshot shows the 'Create Deployment Type Wizard' window. The title bar reads 'Create Deployment Type Wizard'. The left sidebar contains a tree view with the following items: General (selected), General Information, Content, Detection Method, User Experience, Requirements, Dependencies, Summary, Progress, and Completion. The main area is titled 'Specify general information for this deployment type'. Below this title is a paragraph: 'Applications can have any number of deployment types. Deployment types include links to content and settings that specify how the content is delivered.' There are three input fields: 'Name:' with the text 'Adobe Reader 11.0.3 EN 01 PSAppDeployToolkit', 'Administrator comments:' with an empty text box, and 'Languages:' with an empty list box and a 'Select...' button. At the bottom right are four buttons: '< Previous', 'Next >', 'Summary', and 'Cancel'. A help icon (?) is located at the bottom left of the main area.

- Set the content location. Additionally, set the Install and Uninstall programs accordingly. They should be:

```
Deploy-Application.exe -DeploymentType "Install"
```

And:

```
Deploy-Application.exe -DeploymentType "Uninstall"
```

respectively.

Create Deployment Type Wizard

Content

General
General Information
Content
Detection Method
User Experience
Requirements
Dependencies
Summary
Progress
Completion

Specify information about the content to be delivered to target devices

Specify the location of the deployment type's content and other settings that control how content is delivered to target devices. All the contents in the path specified will be delivered.

Content location: Browse...

☐ Persist content in the client cache

☒ Allow clients to share content with other clients on the same subnet

This option allows clients that use Windows BranchCache to download content from on-premises distribution points. Content downloads from cloud-based distribution points can always be shared by clients that use Windows BranchCache.

Specify the command used to install this content.

Installation program: Browse...

Installation start in:

Configuration Manager can remove installations of this content if an uninstall program is specified below.

Uninstall program: Browse...

Uninstall start in:

☐ Run installation and uninstall program as 32-bit process on 64-bit clients.

? < Previous Next > Summary Cancel

- Create a new detection rule. Specify the base MSI product code and modify the Version to be the same as the final version after all patches are installed:

Create a rule that indicates the presence of this application.

Setting Type:

Specify an MSI product code as the basis for this rule.

Product code:

☐ This MSI product code must exist on the target system to indicate presence of this application

☒ This MSI product code must exist on the target system and the following condition must be met to indicate presence of this application:

MSI Property:

Operator:

Value:

- On the User Experience page, ensure you use a combination of settings that allows the user to interact with the application. Failure to do so will result in the application installing silently:

Create Deployment Type Wizard

User Experience

General

General Information

Content

Detection Method

User Experience

Requirements

Dependencies

Summary

Progress

Completion

Specify user experience settings for the application

Installation behavior: Install for system

Logon requirement: Only when a user is logged on

Installation program visibility: Normal

☒ Allow users to view and interact with the program installation

Specify the maximum run time and estimated installation time of the deployment program for this application. The estimated installation time displays to the user when the application installs.

Maximum allowed run time (minutes): 120

Estimated installation time (minutes): 0

< Previous Next > Summary Cancel

- Leave the requirements page blank (or modify according to your environment)
- Leave the software dependencies page blank (or modify according to your environment)
- Accept the defaults to create the Application
- Deploy the Application:

Deploy Software Wizard

General

General

Content

Deployment Settings

Scheduling

User Experience

Alerts

Summary

Progress

Completion

Specify general information for this deployment

Software: Adobe Reader 11.0.3

Collection: SCCM_App_Test

☐ Use default distribution point groups associated to this collection

☒ Automatically distribute content for dependencies

Comments (optional):

- Select the relevant Distribution Points:

Deploy Software Wizard

Content

General
Content
Deployment Settings
Scheduling
User Experience
Alerts
Summary
Progress
Completion

Specify the content destination

Distribution points or distribution point groups that the content has been distributed to:

Name	Type
There are no items to show in this view.	

Additional distribution points, distribution point groups, and the distribution point groups that are currently associated with collections to distribute content to:

Filter...

Name	Description	Associations
	Distribution point	

Add ▼
Remove

? < Previous Next > Summary Cancel

- Configure deployment settings according to whether it should be a mandatory or app catalogue based deployment:

Deploy Software Wizard

Deployment Settings

General
Content
Deployment Settings
Scheduling
User Experience
Alerts
Summary
Progress
Completion

Specify settings to control how this software is deployed


Action:

Purpose:

☐ Pre-deploy software to the user's primary device

☐ Send wake-up packets

☐ Allow clients on a metered Internet connection to download content after the installation deadline, which might incur additional costs



- Specify the deployment schedule:

Deploy Software Wizard

Scheduling

General
Content
Deployment Settings
Scheduling
User Experience
Alerts
Summary
Progress
Completion

Specify the schedule for this deployment

This application will be available as soon as it has been distributed to the content server(s) unless it is scheduled for a later time below. Specify the installation deadline if this is a required application. This deadline is when the application must be installed on the device, including a system restart if necessary.

Time based on: UTC

☐ Schedule the application to be available at:
10/10/2013 1:32 PM

Installation deadline:
☒ As soon as possible after the available time
☐ Schedule at:
10/10/2013 1:32 PM

< Previous Next > Summary Cancel

- Specify User notification settings. In order to prevent excess noise, we recommend only showing notifications for computer restarts:

Deploy Software Wizard

User Experience

General
Content
Deployment Settings
Scheduling
User Experience
Alerts
Summary
Progress
Completion

Specify the user experience for the installation of this software on the selected devices

Specify user experience setting for this deployment

User notifications: Display in Software Center, and only show notifications for computer restarts

When the installation deadline is reached, allow the following activities to be performed outside the maintenance window:

☐ Software Installation

☐ System restart (if required to complete the installation)

Write filter handling for Windows Embedded devices

☒ Commit changes at deadline or during a maintenance window (requires restarts)

If this option is not selected, content will be applied on the overlay and committed later.

< Previous Next > Summary Cancel

- Accept the defaults for the rest of the Deployment (or modify according to your environment)

Example Deployment (Advanced) - Office 2013 SP1

This example is provided as a script with the PSAppDeployToolkit, in the Examples folder. This provides a number of benefits over the standard Microsoft Office Setup Bootstrapper:

- A component based architecture so that core products can be installed, and subsequent components can be installed using the same package with different command-line switches
- The ability to defer the installation up to 3 times

- The ability to close any applications that could cause errors during the installation
- Verification that the required disk space is available
- Full removal of any previous version of Microsoft Office 2007, 2010 or 2013
- Installation of any subsequent patches required after the base installation
- Activation of Microsoft Office components

Note: Office requires a number of modifications in order to install. Please refer to Microsoft's documentation on configuration. This installation script tries to take a lot of work out of the process for you, but you still need to know what you're doing in order to set it up correctly.

The folder structure is laid out as follows:

- Files
 - Office installation files should be placed here
 - Office Configuration MSP created with the Office Customisation Tool should be placed in the *Config* subfolder and be named *Office2013ProPlus.MSP*. Modify the script accordingly if you wish to change. For a basic MSP, you should probably configure Access, Word, Excel and PowerPoint to be the only core applications to install. We can add everything else as components.
 - Customised *Config.xml* file should be edited in *ProPlus.WW* subfolder. At a minimum, you should modify the settings as follows:
 - Display Level = "none"
 - CompletionNotice = "no"
 - SuppressModal = "yes"
 - NoCancel = "yes"
 - AcceptEula = "yes"
 - Security updates and service pack extracted MSPs should be placed in the *Updates* subfolder.
- SupportFiles
 - Contains custom *Config.XML* files which are used to add specific components that might be considered unnecessary in a standard Office install, but could be added later using command-line switches

- Contains Office Scrub tools for Office 2007, 2010 and 2013

Once the folder structure is laid out correctly and the custom `Deploy-Application.ps1` is added (as well as the `PSAppDeployToolkit` package files themselves), the following command-lines are valid:

- Installs Office 2013 with core products

```
Deploy-Application.exe
```

- Installs Office 2013 with core products and InfoPath

```
Deploy-Application.exe -AddInfoPath
```

- Installs InfoPath to an existing Office 2013 installation

```
Deploy-Application.exe -AddComponentsOnly -AddInfoPath
```

Toolkit Reference - Exit Codes

The `PSAppDeployToolkit` has a number of internal exit codes for any issues that may occur.

Exit Code	Description
60000-68999	Reserved for built-in exit codes in <i>Deploy-Application.ps1</i> , <i>Deploy-Application.exe</i> , and <i>AppDeployToolkitMain.ps1</i> .

Exit Code	Description
69000 - 69999	Recommended for user customized exit codes in Deploy-Application.ps1.
70000 - 79999	Recommended for user customized exit codes in Deploy-Application.ps1.
60001	An error occurred in Deploy-Application.ps1. Check your script syntax use.
60002	Administrator privileges required for Execute-ProcessAsUser function.
60003	Failure when loading .NET WinForms / WPF Assemblies.
60004	Failure when displaying the Blocked Application dialog.
60005	AllowSystemInteractionFallback option was not selected in the config XML file, so toolkit will not fall back to SYSTEM context with no interaction.
60006	Failed to export the schedule task XML file in Execute-ProcessAsUser function.
60007	Deploy-Application.ps1 failed to dot source AppDeployToolkitMain.ps1 either because it could not be found or there was an error while it was being dot sourced.
60009	The -UserName parameter in the Execute-ProcessAsUser function has a default value that is empty because no logged in users were detected when the PSAppDeployToolkit was launched.
60010	Deploy-Application.exe failed before <i>PowerShell.exe</i> process could be launched.
60013	If Execute-Process function captures an exit code out of range for int32 then return this custom exit code.

Toolkit Reference - Variables

The PSAppDeployToolkit has a number of internal variables which can be used in your script. Outlined below are each of them:

Toolkit Name

Variable	Description
\$appDeployToolkitName	Short-name of toolkit without spaces
\$appDeployMainScriptFriendlyName	Full name of toolkit including spaces

Script Info

Variable	Description
\$appDeployMainScriptVersion	Version number of the PSAppDeployToolkit
\$appDeployMainScriptMinimumConfigVersion	Minimum version of the config XML file required by the PSAppDeployToolkit
\$appDeployMainScriptDate	Date toolkit was last modified
\$appDeployMainScriptParameters	Contains all parameters and values specified when toolkit was launched

Date, Time & Culture

\$currentDateTime	Current date & time when the PSAppDeployToolkit was launched
\$currentTime	Current time when toolkit was launched
\$currentDate	Current date when toolkit was launched
\$currentTimeZoneBias	TimeZone bias based on the current date / time

\$culture	Object which contains all of the current Windows culture settings
\$currentLanguage	Current Windows two letter ISO language name (e.g. EN, FR, DE, JA etc)
\$currentUILanguage	Current Windows two letter UI ISO language name (e.g. EN, FR, DE, JA etc)

Environment Variables

Path examples are for Windows 7 and higher

Variable	Description
\$envHost	Object that contains details about the current PowerShell console
\$envShellFolders	Object that contains properties from registry path: HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders
\$envAllUsersProfile	%ALLUSERSPROFILE%, e.g. C:\ProgramData
\$envAppData	%APPDATA%, e.g. C:\Users\%USERNAME%\AppData\Roaming
\$envArchitecture	%PROCESSOR_ARCHITECTURE%, e.g. AMD64/IA64/x86. Note - This doesn't tell you the architecture of the processor but only of the current process, so it returns "x86" for a 32-bit WOW process running on 64-bit Windows.
\$envCommonProgramFiles	%COMMONPROGRAMFILES%, e.g. C:\Program Files\Common Files)
\$envCommonProgramFilesX86	%COMMONPROGRAMFILES(x86)%, e.g. C:\Program Files (x86)\Common Files
\$envCommonDesktop	e.g. C:\Users\Public\Desktop
\$envCommonDocuments	e.g. C:\Users\Public\Documents
\$envCommonStartMenuPrograms	e.g. C:\ProgramData\Microsoft\Windows\Start Menu\Programs
\$envCommonStartMenu	e.g. C:\ProgramData\Microsoft\Windows\Start Menu
\$envCommonStartUp	e.g. C:\ProgramData\Microsoft\Windows\Start Menu
\$envCommonTemplates	e.g. C:\ProgramData\Microsoft\Windows\Templates
\$envComputerName	\$COMPUTERNAME%, e.g. Computer1
\$envComputerNameFQDN	Fully qualified computer name, e.g. computer1.contoso.com

Variable	Description
\$envHomeDrive	%HOMEDRIVE%, e.g. c:
\$envHomePath	%HOMEPATH%, e.g. C:\Users\%USERNAME%
\$envHomeShare	%HOMESHARE% (Used instead of HOMEDRIVE if the home folder uses UNC paths.)
\$envLocalAppData	%LOCALAPPDATA%, e.g. C:\Users\%USERNAME%\AppData\Local
\$envLogicalDrives	An array containing all of the logical drives on the system.
\$envProgramFiles	%PROGRAMFILES%, e.g. C:\Program Files
\$envProgramFilesX86	%ProgramFiles(x86)%, e.g. C:\Program Files (x86) (Only on 64 bit. Used to store 32 bit apps.)
\$envProgramData	%PROGRAMDATA%, e.g. C:\ProgramData
\$envPublic	%PUBLIC%, e.g. C:\Users\Public
\$envSystemDrive	%SYSTEMDRIVE%, e.g. c:
\$envSystemRAM	System RAM as an integer
\$envSystemRoot	%SYSTEMROOT%, e.g. C:\Windows
\$envTemp	Checks for the existence of environment variables in the following order and uses the first path found: <ul style="list-style-type: none"> - The path specified by TEMP environment variable, (e.g. C:\Users\%USERNAME%\AppData\Local\Temp). - The path specified by the USERPROFILE environment variable. - The Windows root (C:\Windows) folder.
\$envUserCookies	C:\Users\%USERNAME%\AppData\Local\Microsoft\Windows\INetCookies
\$envUserDesktop	C:\Users\%USERNAME%\Desktop
\$envUserFavorites	C:\Users\%USERNAME%\Favorites
\$envUserInternetCache	C:\Users\%USERNAME%\AppData\Local\Microsoft\Windows\INetCache
\$envUserInternetHistory	C:\Users\%USERNAME%\AppData\Local\Microsoft\Windows\History
\$envUserMyDocuments	C:\Users\%USERNAME%\Documents
\$envUserName	%USERNAME%
\$envUserProfile	%USERPROFILE%, e.g. %SystemDrive%\Users\%USERNAME%
\$envUserSendTo	C:\Users\%USERNAME%\AppData\Roaming\Microsoft\Windows\SendTo
\$envUserStartMenu	C:\Users\%USERNAME%\AppData\Roaming\Microsoft\Windows\Start Menu

Variable	Description
\$envUserStartMenuPrograms	C:\Users\%USERNAME%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs
\$envUserStartUp	C:\Users\%USERNAME%\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup
\$envSystem32Directory	C:\WINDOWS\system32
\$envWinDir	%WINDIR%, e.g. C:\Windows

Domain Membership

Variable	Description
\$IsMachinePartOfDomain	Is machine joined to a domain, e.g. \$true/\$false
\$envMachineWorkgroup	If machine not joined to domain, what is the WORKGROUP it belongs to?
\$envMachineADDomain	Root AD domain name for machine, e.g. domain.contoso.com
\$envLogonServer	FQDN of %LOGONSERVER% used for authenticating logged in user
\$MachineDomainController	FQDN of an AD domain controller used for authentication
\$envMachineDNSDomain	Full Domain name for machine, e.g. <name>.contoso.com
\$envUserDNSDomain	%USERDNSDOMAIN%. Root AD domain name for user, e.g. domain.contoso.com
\$envUserDomain	%USERDOMAIN%, e.g. domain.contoso.com

Operating System

Variable	Description
\$envOS	Object that contains details about the operating system

Variable	Description
\$envOSName	Name of the operating system (e.g. Microsoft Windows 8.1 Pro)
\$envOSServicePack	Latest service pack installed on the system (e.g. Service Pack 3)
\$envOSVersion	Full version number of the OS (e.g. {major}.{minor}.{build}.{revision})
\$envOSVersionMajor	Major portion of the OS version number (e.g. {major}.{minor}.{build}.{revision})
\$envOSVersionMinor	Minor portion of the OS version number (e.g. {major}.{minor}.{build}.{revision})
\$envOSVersionBuild	Build portion of the OS version number (e.g. {major}.{minor}.{build}.{revision})
\$envOSVersionRevision	Revision portion of the OS version number (e.g. {major}.{minor}.{build}.{revision})
\$envOSProductType	OS product type represented as an integer (e.g. 1/2/3)
\$IsServerOS	Is server OS? (e.g. \$true/\$false)
\$IsDomainControllerOS	Is domain controller OS? (e.g. \$true/\$false)
\$IsWorkStationOS	Is workstation OS? (e.g. \$true/\$false)
\$envOSProductTypeName	OS product type name (e.g. Server/Domain Controller/Workstation/Unknown)
\$Is64Bit	Is this a 64-bit OS? (e.g. \$true/\$false)
\$envOSArchitecture	Represents the OS architecture (e.g. 32-Bit/64-Bit)

Current Process Architecture

Variable	Description
\$Is64BitProcess	Is the current process 64-bits? (e.g. \$true/\$false)

Variable	Description
\$psArchitecture	Represents the current process architecture (e.g. x86/x64)

PowerShell And CLR (.NET) Versions

Variable	Description
\$envPSVersionTable	Object containing PowerShell version details from PS variable \$PSVersionTable
\$envPSVersion	Full version number of PS (e.g. {major}.{minor}.{build}.{revision})
\$envPSVersionMajor	Major portion of PS version number (e.g. {major}.{minor}.{build}.{revision})
\$envPSVersionMinor	Minor portion of PS version number (e.g. {major}.{minor}.{build}.{revision})
\$envPSVersionBuild	Build portion of PS version number (e.g. {major}.{minor}.{build}.{revision})
\$envPSVersionRevision	Revision portion of PS version number (e.g. {major}.{minor}.{build}.{revision})
\$envCLRVersion	Full version number of .NET used by PS (e.g. {major}.{minor}.{build}.{revision})
\$envCLRVersionMajor	Major portion of PS .NET version number (e.g. {major}.{minor}.{build}.{revision})
\$envCLRVersionMinor	Minor portion of PS .NET version number (e.g. {major}.{minor}.{build}.{revision})
\$envCLRVersionBuild	Build portion of PS .NET version number (e.g. {major}.{minor}.{build}.{revision})
\$envCLRVersionRevision	Revision portion of PS .NET version number (e.g. {major}.{minor}.{build}.{revision})

Permissions / Accounts

\$CurrentProcessToken	Object that represents the current processes Windows Identity user token. Contains all details regarding user permissions.
\$CurrentProcessSID	Object that represents the current process account SID (e.g. S-1-5-32-544)
\$ProcessNTAccount	Current process NT Account (e.g. NT AUTHORITY\SYSTEM)
\$ProcessNTAccountSID	Current process account SID (e.g. S-1-5-32-544)
\$IsAdmin	Is the current process running with elevated admin privileges? (e.g. \$true/\$false)
\$IsLocalSystemAccount	Is the current process running under the SYSTEM account? (e.g. \$true/\$false)
\$IsLocalServiceAccount	Is the current process running under LOCAL SERVICE account? (e.g. \$true/\$false)
\$IsNetworkServiceAccount	Is the current process running under the NETWORK SERVICE account? (e.g. \$true/\$false)
\$IsServiceAccount	Is the current process running as a service? (e.g. \$true/\$false)
\$IsProcessUserInteractive	Is the current process able to display a user interface?
\$LocalSystemNTAccount	Localized NT account name of the SYSTEM account (e.g. NT AUTHORITY\SYSTEM)
\$SessionZero	Is the current process currently in session zero? In session zero isolation, process is not able to display a user interface. (e.g. \$true/\$false)

Script Name and Script Paths

Variable	Description
\$scriptPath	Fully qualified path of the PSAppDeployToolkit, e.g. C:\Testing\AppDeployToolkit\AppDeployToolkitMain.ps1
\$scriptName	Name of toolkit without file extension e.g. AppDeployToolkitMain
\$scriptFileName	Name of toolkit file e.g. AppDeployToolkitMain.ps1
\$scriptRoot	Path that the PSAppDeployToolkit is located in. e.g. C:\Testing\AppDeployToolkit
\$invokingScript	Fully qualified path of the script that invoked the PSAppDeployToolkit, e.g. C:\Testing\Deploy-Application.ps1
\$scriptParentPath	If the PSAppDeployToolkit was invoked by another script, this contains the path that the invoking script is located in.

PSAppDeployToolkit Dependency Files

Variable	Description
\$appDeployLogoIcon	Path to the logo icon file for the PSAppDeployToolkit, e.g. \$scriptRoot\AppDeployToolkitLogo.ico
\$appDeployLogoBanner	Path to the logo banner file for the PSAppDeployToolkit, e.g. \$scriptRoot\AppDeployToolkitBanner.png
\$appDeployConfigFile	Path to the config XML file for the PSAppDeployToolkit, e.g. \$scriptRoot\AppDeployToolkitConfig.xml
\$appDeployToolkitDotSourceExtensions	Name of the optional extensions file for the PSAppDeployToolkit, e.g. AppDeployToolkitExtensions.ps1

Variable	Description
\$xmlConfigFile	Contains the entire contents of the XML config file
\$configConfigVersion	Version number of the config XML file
\$configConfigDate	Last modified date of the config XML file

Script Directories

Variable	Description
\$dirFiles	Files sub-folder of the PSAppDeployToolkit
\$dirSupportFiles	SupportFiles sub-folder of the PSAppDeployToolkit
\$dirAppDeployTemp	Toolkit temp folder. Configured in XML Config file option Toolkit_TempPath. e.g. Toolkit_TempPath\\${appDeployToolkitName}

Script Naming Conventions

Variable	Description
\$appVendor	Name of the manufacturer that created the package being deployed (e.g. Microsoft)
\$appName	Name of the application being packaged (e.g. Office 2010)
\$appVersion	Version number of the application being packaged (e.g. 14.0)
\$appLang	UI language of the application being packaged (e.g. EN)
\$appRevision	Revision number of the package (e.g. 01)
\$appArch	Architecture of the application being packaged (e.g. x86/x64)
\$installTitle	Combination of the most important details about the application being packaged (e.g. "\$appVendor \$appName \$appVersion")

Variable	Description
\$installName	Combination of any of the following details which were provided: \$appVendor + _ + \$appName + _ + \$appVersion + _ + \$appArch + _ + \$appLang + _ + \$appRevision

Executables

Variable	Description
\$exeWusa	Name of system utility that installs Standalone Windows Updates, e.g. wusa.exe
\$exeMsiexec	Name of system utility that install Windows Installer files, e.g. msiexec.exe
\$exeSchTasks	Path of system utility that allows management of scheduled tasks, e.g. \$envWinDir\System32\schtasks.exe

RegEx Patterns

Variable	Description
\$MSIProductCodeRegExPattern	Contains the regex pattern used to detect a MSI product code.

Registry Keys

Variable	Description
\$regKeyApplications	Array containing the path to the 32-bit and 64-bit portions of the registry that contain information about programs installed on the system. HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall

Variable	Description
\$regKeyLotusNotes	Contains the registry path that stores information about a Lotus Notes installation. - HKLM: SOFTWARE\Lotus\Notes - HKLM: SOFTWARE\Wow6432Node\Lotus\Notes
\$regKeyAppExecution	Contains the registry path where application execution can be blocked by configuring the 'Debugger' value. HKLM: SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options
\$regKeyDeferHistory	The path in the registry where the defer history for the package being installed is stored. \$configToolkitRegPath + \\$appDeployToolkitName\DeferHistory\\${installName}

COM Objects

Variable	Description
\$Shell	Represents and allows use of the WScript.Shell COM object
\$ShellApp	Represents and allows use of the Shell.Application COM object

Log File

Variable	Description
\$logName	Name of the script log file: \${installName} + '_' + \$appDeployToolkitName + '_' + \$deploymentType + '.log'
\$logTempFolder	Temporary log file folder used if the option to compress log files was selected in the config XML file: \$envTemp\\${installName}
\$configToolkitLogDir	Path to log folder defined in XML config file
\$DisableScriptLogging	Dot source this ScriptBlock to disable logging messages to the log file.

Variable	Description
\$RevertScriptLogging	Dot source this ScriptBlock to revert script logging back to its original setting.

Script Parameters

Variable	Description
\$deployAppScriptParameters	Non-default parameters that Deploy-Application.ps1 was launched with
\$appDeployMainScriptParameters	Non-default parameters that AppDeployToolkitMain.ps1 was launched with
\$appDeployExtScriptParameters	Non-default parameters that AppDeployToolkitExtensions.ps1 was launched with

Logged On Users

Variable	Description
\$LoggedOnUserSessions	Object that contains account and session details for all users
\$usersLoggedOn	Array that contains all of the NTAccount names of logged in users
\$CurrentLoggedOnUserSession	Object that contains account and session details for the current process if it is running as a logged in user. This is the object from \$LoggedOnUserSessions where the IsCurrentSession property is \$true.
\$CurrentConsoleUserSession	Objects that contains the account and session details of the console user (user with control of the physical monitor, keyboard, and mouse). This is the object from \$LoggedOnUserSessions where the IsConsoleSession property is \$true.

Variable	Description
\$RunAsActiveUser	The active console user. If no console user exists but users are logged in, such as on terminal servers, then the first logged-in non-console user.

Miscellaneous

Variable	Description
\$dpiPixels	DPI Scale (property only exists if DPI scaling has been changed on the system at least once)
\$runningTaskSequence	Is the current process running in a SCCM task sequence? (e.g. \$true/\$false)
\$IsTaskSchedulerHealthy	Are the task scheduler services in a healthy state? (e.g. \$true/\$false)
\$invalidFileNameChars	Array of all invalid file name characters used to sanitize variables which may be used to create file names.
\$useDefaultMsi	A Zero-Config MSI installation was detected.
\$LocalUsersGroup	Returns the name of the local Users group, typically BUILTIN\Users
\$LocalPowerUsersGroup	Returns the name of the local Power Users group, typically BUILTIN\Power Users group
\$LocalAdministratorsGroup	Returns the name of the local Administrators group, typically BUILTIN\Administrators

Toolkit Reference - Functions

Convert-RegistryPath

SYNOPSIS

Converts the specified registry key path to a format that is compatible with built-in PowerShell cmdlets.

SYNTAX

```
Convert-RegistryPath [-Key] <String> [[-SID] <String>] [[-DisableFunctionLogging]
<Boolean>]
[<CommonParameters>]
```

DESCRIPTION

Converts the specified registry key path to a format that is compatible with built-in PowerShell cmdlets.

Converts registry key hives to their full paths.

Example: HKLM is converted to "Registry::HKEY_LOCAL_MACHINE".

EXAMPLES

EXAMPLE 1

```
Convert-RegistryPath -Key
'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{1AD147D0-
BE0E-3D6C-AC11-64F6DC4163F1}'
```

EXAMPLE 2

```
Convert-RegistryPath -Key
```

```
'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{1AD147D0-BE0E-3D6C-AC11-64F6DC4163F1}'
```

PARAMETERS

-DisableFunctionLogging

Disables logging of this function.

Default: \$true

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 3

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Key

Path to the registry key to convert (can be a registry hive or fully qualified path)

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-SID

The security identifier (SID) for a user.

Specifying this parameter will convert a HKEY_CURRENT_USER registry key to the HKEY_USERS\$SID format.

Specify this parameter from the Invoke-HKCURegistrySettingsForAllUsers function to read/edit HKCU registry settings for all users on the system.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the converted registry key path.

RELATED LINKS

<https://psappdeploytoolkit.com>

Copy-File

SYNOPSIS

Copy a file or group of files to a destination path.

SYNTAX

```
Copy-File [-Path] <String[]> [-Destination] <String> [-Recurse] [-Flatten] [[-ContinueOnError] <Boolean>]  
        [[-ContinueFileCopyOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Copy a file or group of files to a destination path.

EXAMPLES

EXAMPLE 1

```
Copy-File -Path "$dirSupportFiles\MyApp.ini" -Destination "$envWinDir\MyApp.ini"
```

EXAMPLE 2

```
Copy-File -Path "$dirSupportFiles\*.*)" -Destination "$envTemp\tempfiles"
```

Copy all of the files in a folder to a destination folder.

PARAMETERS

-ContinueFileCopyOnError

Continue copying files if an error is encountered.

This will continue the deployment script and will warn about files that failed to be copied.

Default is: \$false.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 4

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-ContinueOnError

Continue if an error is encountered.

This will continue the deployment script, but will not continue copying files if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 3

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Destination

Destination Path of the file to copy.

```
Type: String
Parameter Sets: (All)
Aliases:

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

-Flatten

Flattens the files into the root destination directory.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Path

Path of the file to copy.

Type: String[]
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Recurse

Copy files in subdirectories.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Disable-TerminalServerInstallMode

SYNOPSIS

Changes to user install mode for Remote Desktop Session Host/Citrix servers.

SYNTAX

```
Disable-TerminalServerInstallMode [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Changes to user install mode for Remote Desktop Session Host/Citrix servers.

EXAMPLES

EXAMPLE 1

```
Disable-TerminalServerInstallMode
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Enable-TerminalServerInstallMode

SYNOPSIS

Changes to user install mode for Remote Desktop Session Host/Citrix servers.

SYNTAX

```
Enable-TerminalServerInstallMode [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Changes to user install mode for Remote Desktop Session Host/Citrix servers.

EXAMPLES

EXAMPLE 1

```
Enable-TerminalServerInstallMode
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Execute-MSI

SYNOPSIS

Executes msiexec.exe to perform the following actions for MSI & MSP files and MSI product codes: install, uninstall, patch, repair, active setup.

SYNTAX

```
Execute-MSI [[-Action] <String>] [-Path] <String> [[-Transform] <String>] [[-Parameters] <String>]
    [[-AddParameters] <String>] [-SecureParameters] [[-Patch] <String>] [[-LoggingOptions] <String>]
    [[-private:LogName] <String>] [[-WorkingDirectory] <String>] [-SkipMSIAAlreadyInstalledCheck]
    [-IncludeUpdatesAndHotfixes] [-NoWait] [-PassThru] [[-IgnoreExitCodes] <String>]
    [[-PriorityClass] <ProcessPriorityClass>] [[-ExitOnProcessFailure] <Boolean>] [[-RepairFromSource] <Boolean>]
    [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Executes msiexec.exe to perform the following actions for MSI & MSP files and MSI product codes: install, uninstall, patch, repair, active setup.

If the -Action parameter is set to "Install" and the MSI is already installed, the function will exit.

Sets default switches to be passed to msiexec based on the preferences in the XML configuration file.

Automatically generates a log file name and creates a verbose log file for all msiexec operations.

Expects the MSI or MSP file to be located in the "Files" sub directory of the App Deploy Toolkit.

Expects transform files to be in the same directory as the MSI file.

EXAMPLES

EXAMPLE 1

```
Execute-MSI -Action 'Install' -Path 'Adobe_FlashPlayer_11.2.202.233_x64_EN.msi'
```

Installs an MSI

EXAMPLE 2

```
Execute-MSI -Action 'Install' -Path 'Adobe_FlashPlayer_11.2.202.233_x64_EN.msi' -  
Transform 'Adobe_FlashPlayer_11.2.202.233_x64_EN_01.mst' -Parameters '/QN'
```

Installs an MSI, applying a transform and overriding the default MSI toolkit parameters

EXAMPLE 3

```
[PSObject]$ExecuteMSIResult = Execute-MSI -Action 'Install' -Path  
'Adobe_FlashPlayer_11.2.202.233_x64_EN.msi' -PassThru
```

Installs an MSI and stores the result of the execution into a variable by using the -PassThru option

EXAMPLE 4

```
Execute-MSI -Action 'Uninstall' -Path '{26923b43-4d38-484f-9b9e-de460746276c}'
```

Uninstalls an MSI using a product code

EXAMPLE 5

```
Execute-MSI -Action 'Patch' -Path 'Adobe_Reader_11.0.3_EN.msp'
```

Installs an MSP

PARAMETERS

-Action

The action to perform.

Options: Install, Uninstall, Patch, Repair, ActiveSetup.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: Install
Accept pipeline input: False
Accept wildcard characters: False
```

-AddParameters

Adds to the default parameters specified in the XML configuration file.

Install default is: "REBOOT=ReallySuppress /QB!".

Uninstall default is: "REBOOT=ReallySuppress /QN".

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error occurred while trying to start the process.
Default: \$false.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 14
Default value: False
Accept pipeline input: False
Accept wildcard characters: False
```

-ExitOnProcessFailure

Specifies whether the function should call Exit-Script when the process returns an exit code that is considered an error/failure.
Default: \$true

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 12

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-IgnoreExitCodes

List the exit codes to ignore or * to ignore all exit codes.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: 10

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-IncludeUpdatesAndHotfixes

Include matches against updates and hotfixes in results.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-LoggingOptions

Overrides the default logging options specified in the XML configuration file.
Default options are: "/L*v".

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 7
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-NoWait

Immediately continue after executing the process.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-Parameters

Overrides the default parameters specified in the XML configuration file.

Install default is: "REBOOT=ReallySuppress /QB!".

Uninstall default is: "REBOOT=ReallySuppress /QN".

Type: String

Parameter Sets: (All)

Aliases: Arguments

Required: False

Position: 4

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-PassThru

Returns ExitCode, STDOUT, and STDERR output from the process.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Patch

The name of the patch (msp) file(s) to be applied to the MSI for use with the "Install" action.

The patch file is expected to be in the same directory as the MSI file.

Multiple patches have to be separated by a semi-colon.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Path

The path to the MSI/MSP file or the product code of the installed MSI.

Type: String
Parameter Sets: (All)
Aliases: FilePath

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

-PriorityClass

Specifies priority class for the process.

Options: Idle, Normal, High, AboveNormal, BelowNormal, RealTime.

Default: Normal

Type: ProcessPriorityClass
Parameter Sets: (All)
Aliases:
Accepted values: Normal, Idle, High, RealTime, BelowNormal, AboveNormal

Required: False
Position: 11
Default value: Normal
Accept pipeline input: False
Accept wildcard characters: False

-private:LogName

Type: String
Parameter Sets: (All)
Aliases: LogName

Required: False
Position: 8
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-RepairFromSource

Specifies whether we should repair from source.

Also rewrites local cache.

Default: \$false

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 13
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-SecureParameters

Hides all parameters passed to the MSI or MSP file from the toolkit Log file.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-SkipMSIAlreadyInstalledCheck

Skips the check to determine if the MSI is already installed on the system.
Default is: \$false.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Transform

The name of the transform file(s) to be applied to the MSI.
The transform file is expected to be in the same directory as the MSI file.
Multiple transforms have to be separated by a semi-colon.

Type: String
Parameter Sets: (All)
Aliases:

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

-WorkingDirectory

Overrides the working directory.

The working directory is set to the location of the MSI file.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 9
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PObject

Returns a PObject with the results of the installation

- ExitCode
- STDOUT
- STDERR

RELATED LINKS

<https://psappdeploytoolkit.com>

Execute-MSP

SYNOPSIS

Reads SummaryInfo targeted product codes in MSP file and determines if the MSP file applies to any installed products

If a valid installed product is found, triggers the Execute-MSI function to patch the installation.

Uses default config MSI parameters.

You can use -AddParameters to add additional parameters.

SYNTAX

```
Execute-MSP [-Path] <String> [[-AddParameters] <String>] [<CommonParameters>]
```

EXAMPLES

EXAMPLE 1

```
Execute-MSP -Path 'Adobe_Reader_11.0.3_EN.msp'
```

EXAMPLE 2

```
Execute-MSP -Path 'AcroRdr2017Upd1701130143_MUI.msp' -AddParameters 'ALLUSERS=1'
```

PARAMETERS

-AddParameters

Additional parameters

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Path

Path to the msp file

Type: String

Parameter Sets: (All)

Aliases: FilePath

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Execute-Process

SYNOPSIS

Execute a process with optional arguments, working directory, window style.

SYNTAX

```
Execute-Process [-Path] <String> [[-Parameters] <String[]>] [-SecureParameters]
  [[-WindowStyle] <ProcessWindowStyle>] [-CreateNoWindow] [[-WorkingDirectory]
  <String>] [-NoWait] [-PassThru]
  [-WaitForMsiExec] [[-MsiExecWaitTime] <Int32>] [[-IgnoreExitCodes] <String>]
  [[-PriorityClass] <ProcessPriorityClass>] [[-ExitOnProcessFailure] <Boolean>] [[-
  UseShellExecute] <Boolean>]
  [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Executes a process, e.g.

a file included in the Files directory of the App Deploy Toolkit, or a file on the local machine.

Provides various options for handling the return codes (see Parameters).

EXAMPLES

EXAMPLE 1

```
Execute-Process -Path 'uninstall_flash_player_64bit.exe' -Parameters '/uninstall' -
WindowStyle 'Hidden'
```

If the file is in the "Files" directory of the App Deploy Toolkit, only the file name needs to be specified.

EXAMPLE 2

```
Execute-Process -Path "$dirFiles\Bin\setup.exe" -Parameters '/S' -WindowStyle  
'Hidden'
```

EXAMPLE 3

```
Execute-Process -Path 'setup.exe' -Parameters '/S' -IgnoreExitCodes '1,2'
```

EXAMPLE 4

```
Execute-Process -Path 'setup.exe' -Parameters "-s -  
f2`"$configToolkitLogDir\$installName.log`""
```

Launch InstallShield "setup.exe" from the ".\Files" sub-directory and force log files to the logging folder.

EXAMPLE 5

```
Execute-Process -Path 'setup.exe' -Parameters "/s /v`"ALLUSERS=1 /qn /L*  
\\`"$configToolkitLogDir\$installName.log`""`"
```

Launch InstallShield "setup.exe" with embedded MSI and force log files to the logging folder.

PARAMETERS

-ContinueOnError

Continue if an error occurred while trying to start the process.

Default: \$false.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 10
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-CreateNoWindow

Specifies whether the process should be started with a new window to contain it.

Only works for Console mode applications.

UseShellExecute should be set to \$false.

Default is false.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-ExitOnProcessFailure

Specifies whether the function should call Exit-Script when the process returns an exit code that is considered an error/failure.

Default: \$true

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 8
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-IgnoreExitCodes

List the exit codes to ignore or * to ignore all exit codes.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-MsiExecWaitTime

Specify the length of time in seconds to wait for the msixec engine to become available.

Default: 600 seconds (10 minutes).

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: \$configMSIMutexWaitTime
Accept pipeline input: False
Accept wildcard characters: False

-NoWait

Immediately continue after executing the process.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Parameters

Arguments to be passed to the executable


```
Type: String[]
Parameter Sets: (All)
Aliases: Arguments

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-PassThru

If NoWait is not specified, returns an object with ExitCode, STDOUT and STDERR output from the process.

If NoWait is specified, returns an object with Id, Handle and ProcessName.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Path

Path to the file to be executed.

If the file is located directly in the "Files" directory of the App Deploy Toolkit, only the file name needs to be specified.

Otherwise, the full path of the file must be specified.

If the files is in a subdirectory of "Files", use the "\$dirFiles" variable as shown in the example.

Type: String
Parameter Sets: (All)
Aliases: FilePath

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-PriorityClass

Specifies priority class for the process.

Options: Idle, Normal, High, AboveNormal, BelowNormal, RealTime.

Default: Normal

Type: ProcessPriorityClass
Parameter Sets: (All)
Aliases:
Accepted values: Normal, Idle, High, RealTime, BelowNormal, AboveNormal

Required: False
Position: 7
Default value: Normal
Accept pipeline input: False
Accept wildcard characters: False

-SecureParameters

Hides all parameters passed to the executable from the Toolkit log file

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-UseShellExecute

Specifies whether to use the operating system shell to start the process.

\$true if the shell should be used when starting the process; \$false if the process should be created directly from the executable file.

The word "Shell" in this context refers to a graphical shell (similar to the Windows shell) rather than command shells (for example, bash or sh) and lets users launch graphical applications or open documents.

It lets you open a file or a url and the Shell will figure out the program to open it with.

The WorkingDirectory property behaves differently depending on the value of the UseShellExecute property.

When UseShellExecute is true, the WorkingDirectory property specifies the location of the executable.

When UseShellExecute is false, the WorkingDirectory property is not used to find the executable.

Instead, it is used only by the process that is started and has meaning only within the context of the new process.

If you set UseShellExecute to \$true, there will be no available output from the process.

Default: \$false

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 9
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-WaitForMsiExec

Sometimes an EXE bootstrapper will launch an MSI install.
In such cases, this variable will ensure that
this function waits for the msixec engine to become available before starting the install.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-WindowStyle

Style of the window of the process executed.
Options: Normal, Hidden, Maximized, Minimized.
Default: Normal.
Note: Not all processes honor WindowStyle.
WindowStyle is a recommendation passed to the process.
They can choose to ignore it.

Only works for native Windows GUI applications.

If the WindowStyle is set to Hidden, UseShellExecute should be set to \$true.

```
Type: ProcessWindowStyle
Parameter Sets: (All)
Aliases:
Accepted values: Normal, Hidden, Minimized, Maximized

Required: False
Position: 3
Default value: Normal
Accept pipeline input: False
Accept wildcard characters: False
```

-WorkingDirectory

The working directory used for executing the process.

Defaults to the directory of the file being executed.

Parameter UseShellExecute affects this parameter.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Execute-ProcessAsUser

SYNOPSIS

Execute a process with a logged in user account, by using a scheduled task, to provide interaction with user in the SYSTEM context.

SYNTAX

```
Execute-ProcessAsUser [[-UserName] <String>] [-Path] <String> [[-TempPath]
<String>] [[-Parameters] <String>]
[-SecureParameters] [[-RunLevel] <String>] [-Wait] [-PassThru] [[-
WorkingDirectory] <String>]
[-ContinueOnError] <Boolean> [<CommonParameters>]
```

DESCRIPTION

Execute a process with a logged in user account, by using a scheduled task, to provide interaction with user in the SYSTEM context.

EXAMPLES

EXAMPLE 1

```
Execute-ProcessAsUser -UserName 'CONTOSO\User' -Path "$PSHOME\powershell.exe" -  
Parameters "-Command & { & `"C:\Test\Script.ps1`"; Exit `$LastExitCode }" -Wait
```

Execute process under a user account by specifying a username under which to execute it.

EXAMPLE 2

```
Execute-ProcessAsUser -Path "$PSHOME\powershell.exe" -Parameters "-Command & { &  
`"C:\Test\Script.ps1`"; Exit `$LastExitCode }" -Wait
```

Execute process under a user account by using the default active logged in user that was detected when the toolkit was launched.

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 7

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Parameters

Arguments to be passed to the file being executed.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: 4

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-PassThru

Returns the exit code from this function or the process launched by the scheduled task.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Path

Path to the file being executed.

Type: String
Parameter Sets: (All)
Aliases:

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

-RunLevel

Specifies the level of user rights that Task Scheduler uses to run the task.

The acceptable values for this parameter are:

- HighestAvailable: Tasks run by using the highest available privileges (Admin privileges for Administrators).
Default Value.
- LeastPrivilege: Tasks run by using the least-privileged user account (LUA) privileges.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: HighestAvailable
Accept pipeline input: False
Accept wildcard characters: False
```

-SecureParameters

Hides all parameters passed to the executable from the Toolkit log file.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-TempPath

Path to the temporary directory used to store the script to be executed as user.
If using a user writable directory, ensure you select -RunLevel 'LeastPrivilege'.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-UserName

Logged in Username under which to run the process from.

Default is: The active console user.

If no console user exists but users are logged in, such as on terminal servers, then the first logged-in non-console user.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: $RunAsActiveUser.NTAccount
Accept pipeline input: False
Accept wildcard characters: False
```

-Wait

Wait for the process, launched by the scheduled task, to complete execution before accepting more input.

Default is \$false.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-WorkingDirectory

Set working directory for the process.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: 6

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Int32.

Returns the exit code from this function or the process launched by the scheduled task.

RELATED LINKS

<https://psappdeploytoolkit.com>

Exit-Script

SYNOPSIS

Exit the script, perform cleanup actions, and pass an exit code to the parent process.

SYNTAX

```
Exit-Script [[-ExitCode] <Int32>] [<CommonParameters>]
```

DESCRIPTION

Always use when exiting the script to ensure cleanup actions are performed.

EXAMPLES

EXAMPLE 1

```
Exit-Script
```

EXAMPLE 2

```
Exit-Script -ExitCode 1618
```

PARAMETERS

-ExitCode

The exit code to be passed from the script to the parent process, e.g. SCCM

```
Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: 0
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-FileVersion

SYNOPSIS

Gets the version of the specified file

SYNTAX

```
Get-FileVersion [-File] <String> [-ProductVersion] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Gets the version of the specified file

EXAMPLES

EXAMPLE 1

```
Get-FileVersion -File "$env:ProgramFilesX86\Adobe\Reader 11.0\Reader\AcroRd32.exe"
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-File

Path of the file

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-ProductVersion

Switch that makes the command return ProductVersion instead of FileVersion

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the version of the specified file.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-FreeDiskSpace

SYNOPSIS

Retrieves the free disk space in MB on a particular drive (defaults to system drive)

SYNTAX

```
Get-FreeDiskSpace [[-Drive] <String>] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Retrieves the free disk space in MB on a particular drive (defaults to system drive)

EXAMPLES

EXAMPLE 1

```
Get-FreeDiskSpace -Drive 'C:'
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Drive

Drive to check free disk space on

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: \$env:SystemDrive
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Double

Returns the free disk space in MB

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-HardwarePlatform

SYNOPSIS

Retrieves information about the hardware platform (physical or virtual)

SYNTAX

```
Get-HardwarePlatform [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Retrieves information about the hardware platform (physical or virtual)

EXAMPLES

EXAMPLE 1

```
Get-HardwarePlatform
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the hardware platform (physical or virtual)

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-IniValue

SYNOPSIS

Parses an INI file and returns the value of the specified section and key.

SYNTAX

```
Get-IniValue [-FilePath] <String> [-Section] <String> [-Key] <String> [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Parses an INI file and returns the value of the specified section and key.

EXAMPLES

EXAMPLE 1

```
Get-IniValue -FilePath "$env:ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -  
Key 'KeyFileName'
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-FilePath

Path to the INI file.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Key

Key within the section of the INI file.

Type: String
Parameter Sets: (All)
Aliases:

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

-Section

Section within the INI file.

Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the value of the specified section and key.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-InstalledApplication

SYNOPSIS

Retrieves information about installed applications.

SYNTAX

```
Get-InstalledApplication [[-Name] <String[]>] [-Exact] [-Wildcard] [-Regex] [[-ProductCode] <String>] [-IncludeUpdatesAndHotfixes] [<CommonParameters>]
```

DESCRIPTION

Retrieves information about installed applications by querying the registry.

You can specify an application name, a product code, or both.

Returns information about application publisher, name & version, product code, uninstall string, install source, location, date, and application architecture.

EXAMPLES

EXAMPLE 1

```
Get-InstalledApplication -Name 'Adobe Flash'
```

EXAMPLE 2

```
Get-InstalledApplication -ProductCode '{1AD147D0-BE0E-3D6C-AC11-64F6DC4163F1}'
```

PARAMETERS

-Exact

Specifies that the named application must be matched using the exact name.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-IncludeUpdatesAndHotfixes

Include matches against updates and hotfixes in results.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-Name

The name of the application to retrieve information for.

Performs a contains match on the application display name by default.

Type: String[]

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-ProductCode

The product code of the application to retrieve information for.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-RegEx

Specifies that the named application must be matched using a regular expression search.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-WildCard

Specifies that the named application must be matched using a wildcard search.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PSObject

Returns a PSObject with information about an installed application

- Publisher
- DisplayName
- DisplayVersion
- ProductCode
- UninstallString
- InstallSource
- InstallLocation

- InstallDate
- Architecture

For every detected matching Application the Function puts out a custom Object containing the following Properties:

DisplayName, DisplayVersion, InstallDate, Publisher, Is64BitApplication, ProductCode, InstallLocation, UninstallSubkey, UninstallString, InstallSource.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-LoggedOnUser

SYNOPSIS

Get session details for all local and RDP logged on users.

SYNTAX

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

DESCRIPTION

Get session details for all local and RDP logged on users using Win32 APIs.

Get the following session details:

NTAccount, SID, UserName, DomainName, SessionId, SessionName, ConnectState, IsCurrentSession, IsConsoleSession, IsUserSession, IsActiveUserSession

IsRdpSession, IsLocalAdmin, LogonTime, IdleTime, DisconnectTime, ClientName, ClientProtocolType, ClientDirectory, ClientBuildNumber

EXAMPLES

EXAMPLE 1

```
Get-LoggedInUser
```

PARAMETERS

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

NOTES

Description of ConnectState property:

Value	Description
-------	-------------

Active	A user is logged on to the session.
--------	-------------------------------------

ConnectQuery	The session is in the process of connecting to a client.
--------------	--

Connected	A client is connected to the session.
-----------	---------------------------------------

Disconnected	The session is active, but the client has disconnected from it.
--------------	---

Down The session is down due to an error.
Idle The session is waiting for a client to connect.
Initializing The session is initializing.
Listening The session is listening for connections.
Reset The session is being reset.
Shadowing This session is shadowing another session.

Description of IsActiveUserSession property:

- If a console user exists, then that will be the active user session.
- If no console user exists but users are logged in, such as on terminal servers, then the first logged-in non-console user that has ConnectionState either 'Active' or 'Connected' is the active user.

Description of IsRdpSession property:

- Gets a value indicating whether the user is associated with an RDP client session.

Description of IsLocalAdmin property:

- Checks whether the user is a member of the Administrators group

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-PendingReboot

SYNOPSIS

Get the pending reboot status on a local computer.

SYNTAX

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

DESCRIPTION

Check WMI and the registry to determine if the system has a pending reboot operation from any of the following:

- a) Component Based Servicing (Vista, Windows 2008)
- b) Windows Update / Auto Update (XP, Windows 2003 / 2008)
- c) SCCM 2012 Clients (DetermineIfRebootPending WMI method)
- d) App-V Pending Tasks (global based Appv 5.0 SP2)
- e) Pending File Rename Operations (XP, Windows 2003 / 2008)

EXAMPLES

EXAMPLE 1

```
Get-PendingReboot
```

Returns custom object with following properties:

- ComputerName
- LastBootUpTime
- IsSystemRebootPending
- IsCBServicingRebootPending
- IsWindowsUpdateRebootPending
- IsSCCMClientRebootPending
- IsFileRenameRebootPending
- PendingFileRenameOperations
- ErrorMsg

EXAMPLE 2

```
(Get-PendingReboot).IsSystemRebootPending
```

Returns boolean value determining whether or not there is a pending reboot operation.

PARAMETERS

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PSObject

Returns a custom object with the following properties

- ComputerName
- LastBootUpTime
- IsSystemRebootPending
- IsCBServicingRebootPending
- IsWindowsUpdateRebootPending
- IsSCCMClientRebootPending
- IsFileRenameRebootPending
- PendingFileRenameOperations
- ErrorMessage

NOTES

ErrorMsg only contains something if an error occurred

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-RegistryKey

SYNOPSIS

Retrieves value names and value data for a specified registry key or optionally, a specific value.

SYNTAX

```
Get-RegistryKey [-Key] <String> [[-Value] <String>] [[-SID] <String>] [-  
ReturnEmptyKeyIfExists]  
[-DoNotExpandEnvironmentNames] [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Retrieves value names and value data for a specified registry key or optionally, a specific value.

If the registry key does not exist or contain any values, the function will return \$null by default.

To test for existence of a registry key path, use built-in Test-Path cmdlet.

EXAMPLES

EXAMPLE 1

```
Get-RegistryKey -Key 'HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{1AD147D0-BE0E-3D6C-AC11-64F6DC4163F1}'
```

EXAMPLE 2

```
Get-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\iexplore.exe'
```

EXAMPLE 3

```
Get-RegistryKey -Key 'HKLM:Software\Wow6432Node\Microsoft\Microsoft SQL Server Compact Edition\v3.5' -Value 'Version'
```

EXAMPLE 4

```
Get-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Environment' -Value 'Path' -DoNotExpandEnvironmentNames
```

Returns %ProgramFiles%\Java instead of C:\Program Files\Java

EXAMPLE 5

```
Get-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Example' -Value '(Default)'
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 4

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-DoNotExpandEnvironmentNames

Return unexpanded REG_EXPAND_SZ values.

Default is: \$false.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-Key

Path of the registry key.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ReturnEmptyKeyIfExists

Return the registry key if it exists but it has no property/value pairs underneath it.
Default is: \$false.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-SID

The security identifier (SID) for a user.
Specifying this parameter will convert a HKEY_CURRENT_USER registry key to the HKEY_USERS\$SID format.

Specify this parameter from the Invoke-HKCURegistrySettingsForAllUsers function to read/edit HKCU registry settings for all users on the system.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Value

Value to retrieve (optional).

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the value of the registry key or value.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-SchedulerTask

SYNOPSIS

Retrieve all details for scheduled tasks on the local computer.

SYNTAX

```
Get-SchedulerTask [[-TaskName] <String>] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Retrieve all details for scheduled tasks on the local computer using schtasks.exe.
All property names have spaces and colons removed.

EXAMPLES

EXAMPLE 1

```
Get-SchedulerTask
```

To display a list of all scheduled task properties.

EXAMPLE 2

```
Get-SchedulerTask | Out-GridView
```

To display a grid view of all scheduled task properties.

EXAMPLE 3

```
Get-SchedulerTask | Select-Object -Property TaskName
```

To display a list of all scheduled task names.

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-TaskName

Specify the name of the scheduled task to retrieve details for.
Uses regex match to find scheduled task.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PSObject. This function returns a PSObject with all scheduled task properties.

NOTES

This function has an alias: Get-ScheduledTask if Get-ScheduledTask is not defined

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-ServiceStartMode

SYNOPSIS

Get the service startup mode.

SYNTAX

```
Get-ServiceStartMode [-Name] <String> [[-ComputerName] <String>] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Get the service startup mode.

EXAMPLES

EXAMPLE 1

```
Get-ServiceStartMode -Name 'wuauserv'
```

PARAMETERS

-ComputerName

Specify the name of the computer.

Default is: the local computer.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: $env:ComputerName
Accept pipeline input: False
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Name

Specify the name of the service.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.ServiceProcess.ServiceController.

Returns the service object.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-Shortcut

SYNOPSIS

Get information from a new .lnk or .url type shortcut

SYNTAX

```
Get-Shortcut [-Path] <String> [-ContinueOnError <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Get information from a new .lnk or .url type shortcut.

Returns a hashtable.

EXAMPLES

EXAMPLE 1

```
Get-Shortcut -Path "$env:ProgramData\Microsoft\Windows\Start Menu\My Shortcut.lnk"
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Path

Path to the shortcut to get information from


```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Collections.Hashtable.

Returns a hashtable with the following keys

- TargetPath
- Arguments
- Description
- WorkingDirectory
- WindowStyle
- Hotkey
- IconLocation

- IconIndex
- RunAsAdmin

NOTES

Url shortcuts only support TargetPath, IconLocation and IconIndex.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-UniversalDate

SYNOPSIS

Returns the date/time for the local culture in a universal sortable date time pattern.

SYNTAX

```
Get-UniversalDate [[-DateTime] <String>] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Converts the current datetime or a datetime string for the current culture into a universal sortable date time pattern, e.g.

2013-08-22 11:51:52Z

EXAMPLES

EXAMPLE 1

```
Get-UniversalDate
```

Returns the current date in a universal sortable date time pattern.

EXAMPLE 2

```
Get-UniversalDate -DateTime '25/08/2013'
```

Returns the date for the current culture in a universal sortable date time pattern.

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default: \$false.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-DateTime

Specify the DateTime in the current culture.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: ((Get-Date -Format
(\$culture).DateTimeFormat.UniversalDateTimePattern).ToString())

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the date/time for the local culture in a universal sortable date time pattern.

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-UserProfiles

SYNOPSIS

Get the User Profile Path, User Account Sid, and the User Account Name for all users that log onto the machine and also the Default User (which does not log on).

SYNTAX

```
Get-UserProfiles [[-ExcludeNTAccount] <String[]>] [[-ExcludeSystemProfiles]
<Boolean>] [-ExcludeDefaultUser]
[<CommonParameters>]
```

DESCRIPTION

Get the User Profile Path, User Account Sid, and the User Account Name for all users that log onto the machine and also the Default User (which does not log on).

Please note that the NTAccount property may be empty for some user profiles but the SID and ProfilePath properties will always be populated.

EXAMPLES

EXAMPLE 1

```
Get-UserProfiles
```

Returns the following properties for each user profile on the system: NTAccount, SID, ProfilePath

EXAMPLE 2

```
Get-UserProfiles -ExcludeNTAccount 'CONTOSO\Robot','CONTOSO\ntadmin'
```

EXAMPLE 3

```
[String[]]$ProfilePaths = Get-UserProfiles | Select-Object -ExpandProperty  
'ProfilePath'
```

Returns the user profile path for each user on the system.

This information can then be used to make modifications under the user profile on the filesystem.

PARAMETERS

-ExcludeDefaultUser

Exclude the Default User.

Default is: \$false.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-ExcludeNTAccount

Specify NT account names in Domain\Username format to exclude from the list of user profiles.

```
Type: String[]
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ExcludeSystemProfiles

Exclude system profiles: SYSTEM, LOCAL SERVICE, NETWORK SERVICE.
Default is: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PSObject. Returns a PSObject with the following properties: NTAccount, SID, ProfilePath

RELATED LINKS

<https://psappdeploytoolkit.com>

Get-WindowTitle

SYNOPSIS

Search for an open window title and return details about the window.

SYNTAX

SearchWinTitle

```
Get-WindowTitle -WindowTitle <String> [-DisableFunctionLogging]  
[<CommonParameters>]
```

GetAllWinTitles

```
Get-WindowTitle [-GetAllWindowTitles] [-DisableFunctionLogging]  
[<CommonParameters>]
```


DESCRIPTION

Search for a window title.

If window title searched for returns more than one result, then details for each window will be displayed.

Returns the following properties for each window: WindowTitle, WindowHandle, ParentProcess, ParentProcessMainWindowHandle, ParentProcessId.

Function does not work in SYSTEM context unless launched with "psexec.exe -s -i" to run it as an interactive process under the SYSTEM account.

EXAMPLES

EXAMPLE 1

```
Get-WindowTitle -WindowTitle 'Microsoft Word'
```

Gets details for each window that has the words "Microsoft Word" in the title.

EXAMPLE 2

```
Get-WindowTitle -GetAllWindowTitles
```

Gets details for all windows with a title.

EXAMPLE 3

```
Get-WindowTitle -GetAllWindowTitles | Where-Object { $_.ParentProcess -eq 'WINWORD' }
```

Get details for all windows belonging to Microsoft Word process with name "WINWORD".

PARAMETERS

-DisableFunctionLogging

Disables logging messages to the script log file.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-GetAllWindowsTitles

Get titles for all open windows on the system.

Type: SwitchParameter
Parameter Sets: GetAllWinTitles
Aliases:

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

-WindowTitle

The title of the application window to search for using regex matching.

Type: String

Parameter Sets: SearchWinTitle

Aliases:

Required: True

Position: Named

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Management.Automation.PSObject

Returns a PSObject with the following properties: WindowTitle, WindowHandle, ParentProcess, ParentProcessMainWindowHandle, ParentProcessId.

RELATED LINKS

<https://psappdeploytoolkit.com>

Install-MSUpdates

SYNOPSIS

Install all Microsoft Updates in a given directory.

SYNTAX

```
Install-MSUpdates [-Directory] <String> [<CommonParameters>]
```

DESCRIPTION

Install all Microsoft Updates of type ".exe", ".msu", or ".msp" in a given directory (recursively search directory).

EXAMPLES

EXAMPLE 1

```
Install-MSUpdates -Directory "$dirFiles\MSUpdates"
```

PARAMETERS

-Directory

Directory containing the updates.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Install-SCCMSoftwareUpdates

SYNOPSIS

Scans for outstanding SCCM updates to be installed and installs the pending updates.

SYNTAX

```
Install-SCCMSoftwareUpdates [[-SoftwareUpdatesScanWaitInSeconds] <Int32>]  
    [[-WaitForPendingUpdatesTimeout] <TimeSpan>] [[-ContinueOnError] <Boolean>]  
    [<CommonParameters>]
```

DESCRIPTION

Scans for outstanding SCCM updates to be installed and installs the pending updates.

Only compatible with SCCM 2012 Client or higher.

This function can take several minutes to run.

EXAMPLES

EXAMPLE 1

```
Install-SCCMSoftwareUpdates
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 3

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-SoftwareUpdatesScanWaitInSeconds

The amount of time to wait in seconds for the software updates scan to complete.
Default is: 180 seconds.

Type: Int32

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: 180

Accept pipeline input: False

Accept wildcard characters: False

-WaitForPendingUpdatesTimeout

The amount of time to wait for missing and pending updates to install before exiting the function.
Default is: 45 minutes.

Type: TimeSpan
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: \$(New-TimeSpan -Minutes 45)
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Invoke-HKCURegistrySettingsForAllUsers

SYNOPSIS

Set current user registry settings for all current users and any new users in the future.

SYNTAX

```
Invoke-HKCURegistrySettingsForAllUsers [-RegistrySettings] <ScriptBlock> [[-  
UserProfiles] <PSObject[]>]  
[<CommonParameters>]
```

DESCRIPTION

Set HKCU registry settings for all current and future users by loading their NTUSER.dat registry hive file, and making the modifications.

This function will modify HKCU settings for all users even when executed under the SYSTEM account.

To ensure new users in the future get the registry edits, the Default User registry hive used to provision the registry for new users is modified.

This function can be used as an alternative to using ActiveSetup for registry settings.

The advantage of using this function over ActiveSetup is that a user does not have to log off and log back on before the changes take effect.

EXAMPLES

EXAMPLE 1

```
[ScriptBlock]$HKCURegistrySettings = {
```

```
Set-RegistryKey -Key 'HKCU\Software\Microsoft\Office\14.0\Common' -Name 'qmenable' -  
Value 0 -Type DWord -SID $UserProfile.SID
```

```
Set-RegistryKey -Key 'HKCU\Software\Microsoft\Office\14.0\Common' -Name  
'updatereliabilitydata' -Value 1 -Type DWord -SID $UserProfile.SID
```

```
}
```

```
Invoke-HKCURegistrySettingsForAllUsers -RegistrySettings $HKCURegistrySettings
```

PARAMETERS

-RegistrySettings

Script block which contains HKCU registry settings which should be modified for all users on the system.

Must specify the -SID parameter for all HKCU settings.

```
Type: ScriptBlock  
Parameter Sets: (All)  
Aliases:  
  
Required: True  
Position: 1  
Default value: None  
Accept pipeline input: False  
Accept wildcard characters: False
```

-UserProfiles

Specify the user profiles to modify HKCU registry settings for.

Default is all user profiles except for system profiles.

Type: PSObject[]
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: (Get-UserProfiles)
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Invoke-RegisterOrUnregisterDLL

SYNOPSIS

Register or unregister a DLL file.

SYNTAX

```
Invoke-RegisterOrUnregisterDLL [-FilePath] <String> [[-DLLAction] <String>] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Register or unregister a DLL file using regsvr32.exe.

Function can be invoked using alias: 'Register-DLL' or 'Unregister-DLL'.

EXAMPLES

EXAMPLE 1

```
Register-DLL -FilePath "C:\Test\DcTLSFileToDMSComp.dll"
```

Register DLL file using the "Register-DLL" alias for this function

EXAMPLE 2

```
UnRegister-DLL -FilePath "C:\Test\DcTLSFileToDMSComp.dll"
```

Unregister DLL file using the "Unregister-DLL" alias for this function

EXAMPLE 3

```
Invoke-RegisterOrUnregisterDLL -FilePath "C:\Test\DcTLSFileToDMSComp.dll" -  
DLLAction 'Register'
```

Register DLL file using the actual name of this function

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

```
Type: Boolean  
Parameter Sets: (All)  
Aliases:  
  
Required: False  
Position: 3  
Default value: True  
Accept pipeline input: False  
Accept wildcard characters: False
```

-DLLAction

Specify whether to register or unregister the DLL.

Optional if function is invoked using 'Register-DLL' or 'Unregister-DLL' alias.

Type: String
Parameter Sets: (All)
Aliases:

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

-FilePath

Path to the DLL file.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Invoke-SCCMTask

SYNOPSIS

Triggers SCCM to invoke the requested schedule task id.

SYNTAX

```
Invoke-SCCMTask [-ScheduleID] <String> [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Triggers SCCM to invoke the requested schedule task id.

EXAMPLES

EXAMPLE 1

```
Invoke-SCCMTask 'SoftwareUpdatesScan'
```

EXAMPLE 2

```
Invoke-SCCMTask
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-ScheduleID

Name of the schedule id to trigger.

Options: HardwareInventory, SoftwareInventory, HeartbeatDiscovery, SoftwareInventoryFileCollection, RequestMachinePolicy, EvaluateMachinePolicy, LocationServicesCleanup, SoftwareMeteringReport, SourceUpdate, PolicyAgentCleanup, RequestMachinePolicy2, CertificateMaintenance, PeerDistributionPointStatus,

PeerDistributionPointProvisioning, ComplianceIntervalEnforcement, SoftwareUpdatesAgentAssignmentEvaluation, UploadStateMessage, StateMessageManager, SoftwareUpdatesScan, AMTProvisionCycle, UpdateStorePolicy, StateSystemBulkSend, ApplicationManagerPolicyAction, PowerManagementStartSummarizer

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

New-Folder

SYNOPSIS

Create a new folder.

SYNTAX

```
New-Folder [-Path] <String> [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Create a new folder if it does not exist.

EXAMPLES

EXAMPLE 1

```
New-Folder -Path "$env:WinDir\System32"
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Path

Path to the new folder to create.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

New-MsiTransform

SYNOPSIS

Create a transform file for an MSI database.

SYNTAX

```
New-MsiTransform [-MsiPath] <String> [[-ApplyTransformPath] <String>] [[-NewTransformPath] <String>]
[-TransformProperties] <Hashtable> [[-ContinueOnError] <Boolean>]
[<CommonParameters>]
```

DESCRIPTION

Create a transform file for an MSI database and create/modify properties in the Properties table.

EXAMPLES

EXAMPLE 1

```
[Hashtable]$TransformProperties = {
```

```
'ALLUSERS' = '1'
```

```
'AgreeToLicense' = 'Yes'
```

```
'REBOOT' = 'ReallySuppress'
```

```
'RebootYesNo' = 'No'
```

```
'ROOTDRIVE' = 'C:'
```

```
}
```

```
New-MsiTransform -MsiPath 'C:\Temp\PSADTInstall.msi' -TransformProperties  
$TransformProperties
```

PARAMETERS

-ApplyTransformPath

Specify the path to a transform which should be applied to the MSI database before any new properties are created or modified.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-MsiPath

Specify the path to an MSI file.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-NewTransformPath

Specify the path where the new transform file with the desired properties will be created. If a transform file of the same name already exists, it will be deleted before a new one is created.

Default is: a) If -ApplyTransformPath was specified but not -NewTransformPath, then <ApplyTransformPath>.new.mst
b) If only -MsiPath was specified, then <MsiPath>.mst

Type: String
Parameter Sets: (All)
Aliases:

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

-TransformProperties

Hashtable which contains calls to Set-MsiProperty for configuring the desired properties which should be included in new transform file.

Example hashtable: [Hashtable]\$TransformProperties = @{ 'ALLUSERS' = '1' }

Type: Hashtable
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

New-Shortcut

SYNOPSIS

Creates a new .lnk or .url type shortcut

SYNTAX

```
New-Shortcut [-Path] <String> -TargetPath <String> [-Arguments <String>] [-  
IconLocation <String>]  
[-IconIndex <Int32>] [-Description <String>] [-WorkingDirectory <String>] [-  
WindowStyle <String>]  
[-RunAsAdmin] [-Hotkey <String>] [-ContinueOnError <Boolean>] [<CommonParameters>]
```


DESCRIPTION

Creates a new shortcut .lnk or .url file, with configurable options

EXAMPLES

EXAMPLE 1

```
New-Shortcut -Path "$envProgramData\Microsoft\Windows\Start Menu\My Shortcut.lnk" -  
TargetPath "$envWinDir\System32\notepad.exe" -IconLocation  
"$envWinDir\System32\notepad.exe" -Description 'Notepad' -WorkingDirectory  
"$envHomeDrive\$envHomePath"
```

PARAMETERS

-Arguments

Arguments to be passed to the target path

```
Type: String  
Parameter Sets: (All)  
Aliases:  
  
Required: False  
Position: Named  
Default value: None  
Accept pipeline input: False  
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Description

Description of the shortcut

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-Hotkey

Create a Hotkey to launch the shortcut, e.g.

"CTRL+SHIFT+F"

Type: String
Parameter Sets: (All)
Aliases:

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

-IconIndex

The index of the icon.

Executables, DLLs, ICO files with multiple icons need the icon index to be specified.

This parameter is an Integer.

The first index is 0.

Type: Int32
Parameter Sets: (All)
Aliases:

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

-IconLocation

Location of the icon used for the shortcut

Type: String
Parameter Sets: (All)
Aliases:

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

-Path

Path to save the shortcut

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-RunAsAdmin

Set shortcut to run program as administrator.

This option will prompt user to elevate when executing shortcut.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-TargetPath

Target path or URL that the shortcut launches

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: Named

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-WindowStyle

Windows style of the application.

Options: Normal, Maximized, Minimized.

Default is: Normal.

Type: String
Parameter Sets: (All)
Aliases:

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

-WorkingDirectory

Working Directory to be used for the target path

Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None. This function does not return any output.

NOTES

Url shortcuts only support TargetPath, IconLocation and IconIndex.
Other parameters are ignored.

RELATED LINKS

<https://psappdeploytoolkit.com>

Remove-File

SYNOPSIS

Removes one or more items from a given path on the filesystem.

SYNTAX

Path

```
Remove-File -Path <String[]> [-Recurse] [-ContinueOnError <Boolean>]  
[<CommonParameters>]
```

LiteralPath

```
Remove-File -LiteralPath <String[]> [-Recurse] [-ContinueOnError <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Removes one or more items from a given path on the filesystem.

EXAMPLES

EXAMPLE 1

```
Remove-File -Path 'C:\Windows\Downloaded Program Files\Temp.inf'
```

EXAMPLE 2

```
Remove-File -LiteralPath 'C:\Windows\Downloaded Program Files' -Recurse
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

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

-LiteralPath

Specifies the path on the filesystem to be resolved.

The value of LiteralPath is used exactly as it is typed; no characters are interpreted as wildcards.

Will accept an array of values.

Type: String[]
Parameter Sets: LiteralPath
Aliases:

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

-Path

Specifies the path on the filesystem to be resolved.

The value of Path will accept wildcards.

Will accept an array of values.

Type: String[]
Parameter Sets: Path
Aliases:

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

-Recurse

Deletes the files in the specified location(s) and in all child items of the location(s).

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Remove-Folder

SYNOPSIS

Remove folder and files if they exist.

SYNTAX

```
Remove-Folder [-Path] <String> [-DisableRecursion] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Remove folder and all files with or without recursion in a given path.

EXAMPLES

EXAMPLE 1

```
Remove-Folder -Path "$envWinDir\Downloaded Program Files"
```

Deletes all files and subfolders in the Windows\Downloads Program Files folder

EXAMPLE 2

```
Remove-Folder -Path "$envTemp\MyAppCache" -DisableRecursion
```

Deletes all files in the Temp\MyAppCache folder but does not delete any subfolders.

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-DisableRecursion

Disables recursion while deleting.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Path

Path to the folder to remove.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Remove-InvalidFileNameChars

SYNOPSIS

Remove invalid characters from the supplied string.

SYNTAX

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

DESCRIPTION

Remove invalid characters from the supplied string and returns a valid filename as a string.

EXAMPLES

EXAMPLE 1

```
Remove-InvalidFileNameChars -Name "Filename/\1"
```

PARAMETERS

-Name

Text to remove invalid filename characters from.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: True (ByPropertyName, ByValue)
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

System.String

A string containing invalid filename characters.

OUTPUTS

System.String

Returns the input string with the invalid characters removed.

NOTES

This functions always returns a string however it can be empty if the name only contains invalid characters.

Do no use this command for an entire path as " " is not a valid filename character.

RELATED LINKS

<https://psappdeploytoolkit.com>

Remove-MSIApplications

SYNOPSIS

Removes all MSI applications matching the specified application name.

SYNTAX

```
Remove-MSIApplications [-Name] <String> [-Exact] [-Wildcard] [[-Parameters]
<String>]
    [[-AddParameters] <String>] [[-FilterApplication] <Array>] [[-
ExcludeFromUninstall] <Array>]
    [-IncludeUpdatesAndHotfixes] [[-LoggingOptions] <String>] [[-private:LogName]
<String>] [-PassThru]
    [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```


DESCRIPTION

Removes all MSI applications matching the specified application name.

Enumerates the registry for installed applications matching the specified application name and uninstalls that application using the product code, provided the uninstall string matches "msiexec".

EXAMPLES

EXAMPLE 1

```
Remove-MSIApplications -Name 'Adobe Flash'
```

Removes all versions of software that match the name "Adobe Flash"

EXAMPLE 2

```
Remove-MSIApplications -Name 'Adobe'
```

Removes all versions of software that match the name "Adobe"

EXAMPLE 3

```
Remove-MSIApplications -Name 'Java 8 Update' -FilterApplication @(
```

```
@('Is64BitApplication', $false, 'Exact'),  
    @('Publisher', 'Oracle Corporation', 'Exact')  
)
```

Removes all versions of software that match the name "Java 8 Update" where the software is 32-bits and the publisher is "Oracle Corporation".

EXAMPLE 4

```
Remove-MSIApplications -Name 'Java 8 Update' -FilterApplication @(, @('Publisher',  
'Oracle Corporation', 'Exact')) -ExcludeFromUninstall @(, @('DisplayName', 'Java 8  
Update 45', 'Contains'))
```

Removes all versions of software that match the name "Java 8 Update" and also have "Oracle Corporation" as the Publisher; however, it does not uninstall "Java 8 Update 45" of the software.

NOTE: If only specifying a single row in the two-dimensional arrays, the array must have the extra parentheses and leading comma as in this example.

EXAMPLE 5

```
Remove-MSIApplications -Name 'Java 8 Update' -ExcludeFromUninstall @(,  
@('DisplayName', 'Java 8 Update 45', 'Contains'))
```

Removes all versions of software that match the name "Java 8 Update"; however, it does not uninstall "Java 8 Update 45" of the software.

NOTE: If only specifying a single row in the two-dimensional array, the array must have the extra parentheses and leading comma as in this example.

EXAMPLE 6

```
Remove-MSIApplications -Name 'Java 8 Update' -ExcludeFromUninstall @(
```

```
@('Is64BitApplication', $true, 'Exact'),  
@('DisplayName', 'Java 8 Update 45', 'Exact'),  
@('DisplayName', 'Java 8 Update 4*', 'Wildcard'),  
@('DisplayName', 'Java \d Update \d{3}', 'Regex'),  
@('DisplayName', 'Java 8 Update', 'Contains'))
```

Removes all versions of software that match the name "Java 8 Update"; however, it does not uninstall 64-bit versions of the software, Update 45 of the software, or any Update that starts with 4.

PARAMETERS

-AddParameters

Adds to the default parameters specified in the XML configuration file.
Uninstall default is: "REBOOT=ReallySuppress /QN".

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error occurred while trying to start the processes.
Default: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 8
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-Exact

Specifies that the named application must be matched using the exact name.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-ExcludeFromUninstall

Two-dimensional array that contains one or more (property, value, match-type) sets that should be excluded from uninstall if found.

Properties that can be excluded: ProductCode, DisplayName, DisplayVersion, UninstallString, InstallSource, InstallLocation, InstallDate, Publisher, Is64BitApplication

```
Type: Array
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: @@(())
Accept pipeline input: False
Accept wildcard characters: False
```

-FilterApplication

Two-dimensional array that contains one or more (property, value, match-type) sets that should be used to filter the list of results returned by Get-InstalledApplication to only those that should be uninstalled.

Properties that can be filtered upon: ProductCode, DisplayName, DisplayVersion, UninstallString, InstallSource, InstallLocation, InstallDate, Publisher, Is64BitApplication

```
Type: Array
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: @(())
Accept pipeline input: False
Accept wildcard characters: False
```

-IncludeUpdatesAndHotfixes

Include matches against updates and hotfixes in results.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-LoggingOptions

Overrides the default logging options specified in the XML configuration file.
Default options are: "/L*v".

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Name

The name of the application to uninstall.
Performs a contains match on the application display name by default.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Parameters

Overrides the default parameters specified in the XML configuration file.
Uninstall default is: "REBOOT=ReallySuppress /QN".

Type: String
Parameter Sets: (All)
Aliases: Arguments

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

-PassThru

Returns ExitCode, STDOUT, and STDERR output from the process.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-private:LogName

Type: String
Parameter Sets: (All)
Aliases: LogName

Required: False
Position: 7
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Wildcard

Specifies that the named application must be matched using a wildcard search.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

PSObject

Returns an object with the following properties:

- ExitCode
- StdOut
- StdErr

NOTES

More reading on how to create arrays if having trouble with -FilterApplication or -ExcludeFromUninstall parameter: <http://blogs.msdn.com/b/powershell/archive/2007/01/23/array-literals-in-powershell.aspx>

RELATED LINKS

<https://psappdeploytoolkit.com>

Remove-RegistryKey

SYNOPSIS

Deletes the specified registry key or value.

SYNTAX

```
Remove-RegistryKey [-Key] <String> [[-Name] <String>] [-Recurse] [[-SID] <String>]  
[[ -ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Deletes the specified registry key or value.

EXAMPLES

EXAMPLE 1

```
Remove-RegistryKey -Key  
'HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce'
```

EXAMPLE 2

```
Remove-RegistryKey -Key 'HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Run' -Name  
'RunAppInstall'
```

EXAMPLE 3

```
Remove-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Example' -Name  
'(Default)'
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 4

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Key

Path of the registry key to delete.

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-Name

Name of the registry value to delete.

Type: String
Parameter Sets: (All)
Aliases:

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

-Recurse

Delete registry key recursively.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-SID

The security identifier (SID) for a user.

Specifying this parameter will convert a HKEY_CURRENT_USER registry key to the HKEY_USERS\$SID format.

Specify this parameter from the Invoke-HKCURegistrySettingsForAllUsers function to read/edit HKCU registry settings for all users on the system.

Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Resolve-Error

SYNOPSIS

Enumerate error record details.

SYNTAX

```
Resolve-Error [[-ErrorRecord] <Array>] [[-Property] <String[]>] [-GetErrorRecord]  
[-GetErrorInvocation]  
[-GetErrorException] [-GetErrorInnerException] [<CommonParameters>]
```

DESCRIPTION

Enumerate an error record, or a collection of error record, properties.
By default, the details for the last error will be enumerated.

EXAMPLES

EXAMPLE 1

```
Resolve-Error
```

EXAMPLE 2

```
Resolve-Error -Property *
```

EXAMPLE 3

```
Resolve-Error -Property InnerException
```

EXAMPLE 4

```
Resolve-Error -GetErrorInvocation:$false
```

PARAMETERS

-ErrorRecord

The error record to resolve.

The default error record is the latest one: `$global:Error[0]`.

This parameter will also accept an array of error records.

Type: Array

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: None

Accept pipeline input: True (ByPropertyName, ByValue)

Accept wildcard characters: False

-GetErrorException

Get error record exception details as represented by `$_Exception`.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-GetErrorInnerException

Get error record inner exception details as represented by `$_Exception.InnerException`. Will retrieve all inner exceptions if there is more than one.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-GetErrorInvocation

Get error record invocation information as represented by `$_InvocationInfo`.


```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-GetErrorRecord

Get error record details as represented by \$_.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-Property

The list of properties to display from the error record.

Use "*" to display all properties.

Default list of error properties is: Message, FullyQualifiedErrorId, ScriptStackTrace, PositionMessage, InnerException

Type: String[]
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: ('Message', 'InnerException', 'FullyQualifiedErrorId',
'ScriptStackTrace', 'PositionMessage')
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

System.Array.

Accepts an array of error records.

OUTPUTS

System.String

Displays the error record details.

RELATED LINKS

<https://psappdeploytoolkit.com>

Send-Keys

SYNOPSIS

Send a sequence of keys to one or more application windows.

SYNTAX

```
Send-Keys [[-WindowTitle] <String>] [-GetAllWindowTitles] [[-WindowHandle]
<IntPtr>] [[-Keys] <String>]
    [[-WaitSeconds] <Int32>] [<CommonParameters>]
```

DESCRIPTION

Send a sequence of keys to one or more application window.

If window title searched for returns more than one window, then all of them will receive the sent keys.

Function does not work in SYSTEM context unless launched with "psexec.exe -s -i" to run it as an interactive process under the SYSTEM account.

EXAMPLES

EXAMPLE 1

```
Send-Keys -WindowTitle 'foobar - Notepad' -Key 'Hello world'
```

Send the sequence of keys "Hello world" to the application titled "foobar - Notepad".

EXAMPLE 2

```
Send-Keys -WindowTitle 'foobar - Notepad' -Key 'Hello world' -WaitSeconds 5
```

Send the sequence of keys "Hello world" to the application titled "foobar - Notepad" and wait 5 seconds.

EXAMPLE 3

```
Send-Keys -WindowHandle ([IntPtr]17368294) -Key 'Hello world'
```

Send the sequence of keys "Hello world" to the application with a Window Handle of '17368294'.

PARAMETERS

-GetAllWindowsTitles

Get titles for all open windows on the system.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: False
Accept pipeline input: False
Accept wildcard characters: False
```

-Keys

The sequence of keys to send.

Info on Key input at: [http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys(v=vs.100).aspx)

Type: String
Parameter Sets: (All)
Aliases:

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

-WaitSeconds

An optional number of seconds to wait after the sending of the keys.

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: 0
Accept pipeline input: False
Accept wildcard characters: False

-WindowHandle

Send keys to a specific window where the Window Handle is already known.

Type: IntPtr
Parameter Sets: (All)
Aliases:

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

-WindowTitle

The title of the application window to search for using regex matching.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

[http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys\(v=vs.100\).aspx](http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys(v=vs.100).aspx)
[X](#)

[https://psappdeploytoolkit.com\]\(http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys\(v=vs.100\).aspx](https://psappdeploytoolkit.com](http://msdn.microsoft.com/en-us/library/System.Windows.Forms.SendKeys(v=vs.100).aspx)

<https://psappdeploytoolkit.com>)

Set-ActiveSetup

SYNOPSIS

Creates an Active Setup entry in the registry to execute a file for each user upon login.

SYNTAX

Create

```
Set-ActiveSetup -StubExePath <String> [-Arguments <String>] [-Description <String>]
[-Key <String>]
[-Version <String>] [-Locale <String>] [-DisableActiveSetup] [-
ExecuteForCurrentUser <Boolean>]
[-ContinueOnError <Boolean>] [<CommonParameters>]
```

Purge

```
Set-ActiveSetup [-Key <String>] [-PurgeActiveSetupKey] [-ContinueOnError <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Active Setup allows handling of per-user changes registry/file changes upon login.

A registry key is created in the HKLM registry hive which gets replicated to the HKCU hive when a user logs in.

If the "Version" value of the Active Setup entry in HKLM is higher than the version value in HKCU, the file referenced in "StubPath" is executed.

This Function:

- Creates the registry entries in HKLM:SOFTWARE\Microsoft\Active Setup\Installed Components\$installName.
- Creates StubPath value depending on the file extension of the \$StubExePath parameter.
- Handles Version value with YYYYMMDDHHMMSS granularity to permit re-installs on the same day and still trigger Active Setup after Version increase.
- Copies/overwrites the StubPath file to \$StubExePath destination path if file exists in 'Files' subdirectory of script directory.
- Executes the StubPath file for the current user based on \$ExecuteForCurrentUser (no need to logout/login to trigger Active Setup).

EXAMPLES

EXAMPLE 1

```
Set-ActiveSetup -StubExePath 'C:\Users\Public\Company\ProgramUserConfig.vbs' -  
Arguments '/Silent' -Description 'Program User Config' -Key 'ProgramUserConfig' -  
Locale 'en'
```


EXAMPLE 2

```
Set-ActiveSetup -StubExePath "$env:WinDir\regedit.exe" -Arguments "/S`n`"%SystemDrive%\Program Files (x86)\PS App Deploy\PSAppDeployHKCUSettings.reg`"" -Description 'PS App Deploy Config' -Key 'PS_App_Deploy_Config' -ContinueOnError $true
```

EXAMPLE 3

```
Set-ActiveSetup -Key 'ProgramUserConfig' -PurgeActiveSetupKey
```

Deletes "ProgramUserConfig" active setup entry from all registry hives.

PARAMETERS

-Arguments

Arguments to pass to the file being executed.

```
Type: String
Parameter Sets: Create
Aliases:

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

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Description

Description for the Active Setup.

Users will see "Setting up personalized settings for: \$Description" at logon.

Default is: \$installName.

Type: String

Parameter Sets: Create

Aliases:

Required: False

Position: Named

Default value: \$installName

Accept pipeline input: False

Accept wildcard characters: False

-DisableActiveSetup

Disables the Active Setup entry so that the StubPath file will not be executed.
This also disables -ExecuteForCurrentUser

```
Type: SwitchParameter
Parameter Sets: Create
Aliases:

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

-ExecuteForCurrentUser

Specifies whether the StubExePath should be executed for the current user.
Since this user is already logged in, the user won't have the application started without logging out and logging back in.
Default: \$true

```
Type: Boolean
Parameter Sets: Create
Aliases:

Required: False
Position: Named
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-Key

Name of the registry key for the Active Setup entry.

Default is: \$installName.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: Named
Default value: $installName
Accept pipeline input: False
Accept wildcard characters: False
```

-Locale

Optional.

Arbitrary string used to specify the installation language of the file being executed.

Not replicated to HKCU.

```
Type: String
Parameter Sets: Create
Aliases:

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

-PurgeActiveSetupKey

Remove Active Setup entry from HKLM registry hive.

Will also load each logon user's HKCU registry hive to remove Active Setup entry.

Function returns after purging.

```
Type: SwitchParameter
Parameter Sets: Purge
Aliases:

Required: True
Position: Named
Default value: False
Accept pipeline input: False
Accept wildcard characters: False
```

-StubExePath

Full destination path to the file that will be executed for each user that logs in.

If this file exists in the 'Files' subdirectory of the script directory, it will be copied to the destination path.

```
Type: String
Parameter Sets: Create
Aliases:

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

-Version

Optional.

Specify version for Active setup entry.

Active Setup is not triggered if Version value has more than 8 consecutive digits.

Use commas to get around this limitation.

Default: YYYYMMDDHHMMSS

Type: String

Parameter Sets: Create

Aliases:

Required: False

Position: Named

Default value: ((Get-Date -Format 'yyMM,ddHH,mmss').ToString())

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Boolean

Returns \$true if Active Setup entry was created or updated, \$false if Active Setup entry was not created or updated.

NOTES

Original code borrowed from: Denis St-Pierre (Ottawa, Canada), Todd MacNaught (Ottawa, Canada)

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-IniValue

SYNOPSIS

Opens an INI file and sets the value of the specified section and key.

SYNTAX

```
Set-IniValue [-FilePath] <String> [-Section] <String> [-Key] <String> [-Value]
<Object>
[[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Opens an INI file and sets the value of the specified section and key.

EXAMPLES

EXAMPLE 1

```
Set-IniValue -FilePath "$env:ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -
Key 'KeyFileName' -Value 'MyFile.ID'
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 5

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-FilePath

Path to the INI file.

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-Key

Key within the section of the INI file.

```
Type: String
Parameter Sets: (All)
Aliases:

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

-Section

Section within the INI file.

```
Type: String
Parameter Sets: (All)
Aliases:

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

-Value

Value for the key within the section of the INI file.

To remove a value, set this variable to \$null.

Type: Object

Parameter Sets: (All)

Aliases:

Required: True

Position: 4

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-ItemPermission

SYNOPSIS

Allow you to easily change permissions on files or folders

SYNTAX

EnableInheritance

```
Set-ItemPermission [-Path] <String> [-EnableInheritance] [<CommonParameters>]
```

DisableInheritance

```
Set-ItemPermission [-Path] <String> [-User] <String[]> [-Permission] <String[]> [[-PermissionType] <String>]  
    [[-Inheritance] <String[]>] [[-Propagation] <String>] [[-Method] <String>]  
    [<CommonParameters>]
```

EXAMPLES

EXAMPLE 1

Will grant FullControl permissions to 'John' and 'Users' on 'C:\Temp' and its files and folders children.

```
PS C:\>Set-ItemPermission -Path 'C:\Temp' -User 'DOMAIN\John', 'BUILTIN\Utilisateurs' -  
Permission FullControl -Inheritance ObjectInherit,ContainerInherit
```

EXAMPLE 2

Will grant Read permissions to 'John' on 'C:\Temp\pic.png'

```
PS C:\>Set-ItemPermission -Path 'C:\Temp\pic.png' -User 'DOMAIN\John' -Permission 'Read'
```

EXAMPLE 3

Will remove all permissions to 'John' on 'C:\Temp\Private'

```
PS C:\>Set-ItemPermission -Path 'C:\Temp\Private' -User 'DOMAIN\John' -Permission 'None' -Method 'RemoveAll'
```

PARAMETERS

-EnableInheritance

Enables inheritance on the files/folders.

Type: SwitchParameter
Parameter Sets: EnableInheritance
Aliases:

Required: True
Position: 2
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-Inheritance

Sets permission inheritance.

Does not apply to files.

Multiple options can be specified.

Allowed options: ObjectInherit, ContainerInherit, None Default: None

None - The permission entry is not inherited by child objects, ObjectInherit - The permission entry is inherited by child leaf objects.

ContainerInherit - The permission entry is inherited by child container objects.

```
Type: String[]
Parameter Sets: DisableInheritance
Aliases:

Required: False
Position: 5
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Method

Specifies which method will be used to apply the permissions.

Allowed options: Add, Set, Reset.

Add - adds permissions rules but it does not remove previous permissions, Set - overwrites matching permission rules with new ones, Reset - removes matching permissions rules and then adds permission rules, Remove - Removes matching permission rules, RemoveSpecific - Removes specific permissions, RemoveAll - Removes all permission rules for specified user/s

Default: Add

Type: String
Parameter Sets: DisableInheritance
Aliases: ApplyMethod, ApplicationMethod

Required: False
Position: 7
Default value: Add
Accept pipeline input: False
Accept wildcard characters: False

-Path

Path to the folder or file you want to modify (ex: C:\Temp)

Type: String
Parameter Sets: (All)
Aliases: File, Folder

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Permission

Permission or list of permissions to be set/added/removed/replaced.

To see all the possible permissions go to '<http://technet.microsoft.com/fr-fr/library/ff730951.aspx>'.

Permission DeleteSubdirectoriesAndFiles does not apply to files.

Type: String[]
Parameter Sets: DisableInheritance
Aliases: Acl, Grant, Permissions, Deny

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

-PermissionType

Sets Access Control Type of the permissions.
Allowed options: Allow, Deny Default: Allow

Type: String
Parameter Sets: DisableInheritance
Aliases: AccessControlType

Required: False
Position: 4
Default value: Allow
Accept pipeline input: False
Accept wildcard characters: False

-Propagation

Sets how to propagate inheritance.
Does not apply to files.
Allowed options: None, InheritOnly, NoPropagateInherit Default: None

None - Specifies that no inheritance flags are set.

NoPropagateInherit - Specifies that the permission entry is not propagated to child objects.

InheritOnly - Specifies that the permission entry is propagated only to child objects.
This includes both container and leaf child objects.

Type: String
Parameter Sets: DisableInheritance
Aliases:

Required: False
Position: 6
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-User

One or more user names (ex: BUILTIN\Users, DOMAIN\Admin) to give the permissions to. If you want to use SID, prefix it with an asterisk * (ex: *S-1-5-18)

Type: String[]
Parameter Sets: DisableInheritance
Aliases: Username, Users, SID, Usernames

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

NOTES

Original Author: Julian DA CUNHA - dacunha.julian@gmail.com, used with permission

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-PinnedApplication

SYNOPSIS

Pins or unpins a shortcut to the start menu or task bar.

SYNTAX

```
Set-PinnedApplication [-Action] <String> [-FilePath] <String> [<CommonParameters>]
```

DESCRIPTION

Pins or unpins a shortcut to the start menu or task bar.

This should typically be run in the user context, as pinned items are stored in the user profile.

EXAMPLES

EXAMPLE 1

```
Set-PinnedApplication -Action 'PinToStartMenu' -FilePath  
"$envProgramFilesX86\IBM\Lotus\Notes\notes.exe"
```

EXAMPLE 2

```
Set-PinnedApplication -Action 'UnpinFromTaskbar' -FilePath  
"$envProgramFilesX86\IBM\Lotus\Notes\notes.exe"
```

PARAMETERS

-Action

Action to be performed.

Options: 'PinToStartMenu','UnpinFromStartMenu','PinToTaskbar','UnpinFromTaskbar'.

```
Type: String  
Parameter Sets: (All)  
Aliases:  
  
Required: True  
Position: 1  
Default value: None  
Accept pipeline input: False  
Accept wildcard characters: False
```

-FilePath

Path to the shortcut file to be pinned or unpinned.

```
Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

NOTES

Windows 10 logic borrowed from Stuart Pearson (<https://pinto10blog.wordpress.com/2016/09/10/pinto10/>)

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-RegistryKey

SYNOPSIS

Creates a registry key name, value, and value data; it sets the same if it already exists.

SYNTAX

```
Set-RegistryKey [-Key] <String> [[-Name] <String>] [[-Value] <Object>] [[-Type]
<RegistryValueKind>]
[[ -SID] <String>] [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Creates a registry key name, value, and value data; it sets the same if it already exists.

EXAMPLES

EXAMPLE 1

```
Set-RegistryKey -Key $blockedAppPath -Name 'Debugger' -Value
$blockedAppDebuggerValue
```

EXAMPLE 2

```
Set-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SOFTWARE' -Name 'Application' -Type
'DWord' -Value '1'
```

EXAMPLE 3

```
Set-RegistryKey -Key  
'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce' -Name  
'Debugger' -Value $blockedAppDebuggerValue -Type String
```

EXAMPLE 4

```
Set-RegistryKey -Key 'HKCU\Software\Microsoft\Example' -Name 'Data' -Value  
(0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x02,0x01,0x01,0x01,0x  
01,0x01,0x01,0x01,0x02,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x00,0x01,0x01,0x01,0x02,  
0x02,0x02) -Type 'Binary'
```

EXAMPLE 5

```
Set-RegistryKey -Key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Example' -Name  
'(Default)' -Value "Text"
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Key

The registry key path.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Name

The value name.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-SID

The security identifier (SID) for a user.

Specifying this parameter will convert a HKEY_CURRENT_USER registry key to the HKEY_USERS\$SID format.

Specify this parameter from the Invoke-HKCURegistrySettingsForAllUsers function to read/edit HKCU registry settings for all users on the system.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Type

The type of registry value to create or set.

Options: 'Binary','DWord','ExpandString','MultiString','None','QWord','String','Unknown'.

Default: String.

DWord should be specified as a decimal.

Type: RegistryValueKind

Parameter Sets: (All)

Aliases:

Accepted values: Unknown, String, ExpandString, Binary, DWord, MultiString, QWord, None

Required: False

Position: 4

Default value: String

Accept pipeline input: False

Accept wildcard characters: False

-Value

The value data.

Type: Object

Parameter Sets: (All)

Aliases:

Required: False

Position: 3

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-ServiceStartMode

SYNOPSIS

Set the service startup mode.

SYNTAX

```
Set-ServiceStartMode [-Name] <String> [[-ComputerName] <String>] [-StartMode]
<String>
[[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Set the service startup mode.

EXAMPLES

EXAMPLE 1

```
Set-ServiceStartMode -Name 'wuauserv' -StartMode 'Automatic (Delayed Start)'
```

PARAMETERS

-ComputerName

Specify the name of the computer.

Default is: the local computer.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: $env:ComputerName
Accept pipeline input: False
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 4

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Name

Specify the name of the service.

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-StartMode

Specify startup mode for the service.

Options: Automatic, Automatic (Delayed Start), Manual, Disabled, Boot, System.

Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Set-Shortcut

SYNOPSIS

Modifies a .lnk or .url type shortcut

SYNTAX

Default (Default)

```
Set-Shortcut [-Path] <String> [-TargetPath <String>] [-Arguments <String>] [-  
IconLocation <String>]  
    [-IconIndex <String>] [-Description <String>] [-WorkingDirectory <String>] [-  
WindowStyle <String>]  
    [-RunAsAdmin <Boolean>] [-Hotkey <String>] [-ContinueOnError <Boolean>]  
    [<CommonParameters>]
```

Pipeline

```
Set-Shortcut [-PathHash] <Hashtable> [-TargetPath <String>] [-Arguments <String>]  
    [-IconLocation <String>]  
    [-IconIndex <String>] [-Description <String>] [-WorkingDirectory <String>] [-  
WindowStyle <String>]  
    [-RunAsAdmin <Boolean>] [-Hotkey <String>] [-ContinueOnError <Boolean>]  
    [<CommonParameters>]
```

DESCRIPTION

Modifies a shortcut - .lnk or .url file, with configurable options.

Only specify the parameters that you want to change.

EXAMPLES

EXAMPLE 1

```
Set-Shortcut -Path "$envProgramData\Microsoft\Windows\Start Menu\My Shortcut.lnk" -  
TargetPath "$envWinDir\System32\notepad.exe" -IconLocation  
"$envWinDir\System32\notepad.exe" -IconIndex 0 -Description 'Notepad' -  
WorkingDirectory "$envHomeDrive\$envHomePath"
```

PARAMETERS

-Arguments

Changes Arguments to be passed to the target path

```
Type: String  
Parameter Sets: (All)  
Aliases:  
  
Required: False  
Position: Named  
Default value: None  
Accept pipeline input: False  
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-Description

Changes description of the shortcut

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

-Hotkey

Changes the Hotkey to launch the shortcut, e.g.

"CTRL+SHIFT+F"

Type: String
Parameter Sets: (All)
Aliases:

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

-IconIndex

Change the index of the icon.

Executables, DLLs, ICO files with multiple icons need the icon index to be specified.

This parameter is an Integer.

The first index is 0.

Type: String
Parameter Sets: (All)
Aliases:

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

-IconLocation

Changes location of the icon used for the shortcut

Type: String
Parameter Sets: (All)
Aliases:

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

-Path

Path to the shortcut to be changed

Type: String
Parameter Sets: Default
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: True (ByValue)
Accept wildcard characters: False

-PathHash

Type: Hashtable
Parameter Sets: Pipeline
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: True (ByValue)
Accept wildcard characters: False

-RunAsAdmin

Set shortcut to run program as administrator.
This option will prompt user to elevate when executing shortcut.
If not specified or set to \$null, the flag will not be changed.

Type: Boolean
Parameter Sets: (All)
Aliases:

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

-TargetPath

Changes target path or URL that the shortcut launches

Type: String
Parameter Sets: (All)
Aliases:

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

-WindowState

Changes the Windows style of the application.

Options: Normal, Maximized, Minimized, DontChange.

Default is: DontChange.

Type: String
Parameter Sets: (All)
Aliases:

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

-WorkingDirectory

Changes Working Directory to be used for the target path

Type: String
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

PSObject

Path to the shortcut to be changed or a hashtable of parameters to be changed

OUTPUTS

None

This function does not generate any output.

NOTES

Url shortcuts only support TargetPath, IconLocation and IconIndex.
Other parameters are ignored.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-BalloonTip

SYNOPSIS

Displays a balloon tip notification in the system tray.

SYNTAX

```
Show-BalloonTip [-BalloonTipText] <String> [[-BalloonTipTitle] <String>] [[-BalloonTipIcon] <ToolTipIcon>] [[-BalloonTipTime] <Int32>] [-NoWait] [<CommonParameters>]
```

DESCRIPTION

Displays a balloon tip notification in the system tray.

EXAMPLES

EXAMPLE 1

```
Show-BalloonTip -BalloonTipText 'Installation Started' -BalloonTipTitle 'Application Name'
```

EXAMPLE 2

```
Show-BalloonTip -BalloonTipIcon 'Info' -BalloonTipText 'Installation Started' -BalloonTipTitle 'Application Name' -BalloonTipTime 1000
```

PARAMETERS

-BalloonTipIcon

Icon to be used.

Options: 'Error', 'Info', 'None', 'Warning'.

Default is: Info.

```
Type: ToolTipIcon
Parameter Sets: (All)
Aliases:
Accepted values: None, Info, Warning, Error

Required: False
Position: 3
Default value: Info
Accept pipeline input: False
Accept wildcard characters: False
```

-BalloonTipText

Text of the balloon tip.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-BalloonTipTime

Time in milliseconds to display the balloon tip.

Default: 10000.

```
Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: 10000
Accept pipeline input: False
Accept wildcard characters: False
```

-BalloonTipTitle

Title of the balloon tip.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: $installTitle
Accept pipeline input: False
Accept wildcard characters: False
```

-NoWait

Create the balloontip asynchronously.

Default: \$false

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: 5

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the version of the specified file.

NOTES

For Windows 10 OS and above a Toast notification is displayed in place of a balloon tip if toast notifications are enabled in the XML config file.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-DialogBox

SYNOPSIS

Display a custom dialog box with optional title, buttons, icon and timeout.

Show-InstallationPrompt is recommended over this function as it provides more customization and uses consistent branding with the other UI components.

SYNTAX

```
Show-DialogBox [-Text] <String> [-Title <String>] [-Buttons <String>] [-  
DefaultButton <String>]  
[-Icon <String>] [-Timeout <String>] [-TopMost <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Display a custom dialog box with optional title, buttons, icon and timeout.

The default button is "OK", the default Icon is "None", and the default Timeout is None

EXAMPLES

EXAMPLE 1

```
Show-DialogBox -Title 'Installed Complete' -Text 'Installation has completed.  
Please click OK and restart your computer.' -Icon 'Information'
```

EXAMPLE 2

```
Show-DialogBox -Title 'Installation Notice' -Text 'Installation will take  
approximately 30 minutes. Do you wish to proceed?' -Buttons 'OKCancel' -  
DefaultButton 'Second' -Icon 'Exclamation' -Timeout 600 -Topmost $false
```

PARAMETERS

-Buttons

Buttons to be included on the dialog box.

Options: OK, OKCancel, AbortRetryIgnore, YesNoCancel, YesNo, RetryCancel, CancelTryAgainContinue.

Default: OK.

Type: String

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: OK

Accept pipeline input: False

Accept wildcard characters: False

-DefaultButton

The Default button that is selected.

Options: First, Second, Third.

Default: First.

Type: String
Parameter Sets: (All)
Aliases:

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

-Icon

Icon to display on the dialog box.

Options: None, Stop, Question, Exclamation, Information.

Default: None

Type: String
Parameter Sets: (All)
Aliases:

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

-Text

Text in the message dialog box

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Timeout

Timeout period in seconds before automatically closing the dialog box with the return message "Timeout".

Default: UI timeout value set in the config XML file.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: Named
Default value: \$configInstallationUITimeout
Accept pipeline input: False
Accept wildcard characters: False

-Title

Title of the message dialog box

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: Named
Default value: \$installTitle
Accept pipeline input: False
Accept wildcard characters: False

-TopMost

Specifies whether the message box is a system modal message box and appears in a topmost window.

Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the text of the button that was clicked.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-InstallationProgress

SYNOPSIS

Displays a progress dialog in a separate thread with an updateable custom message.

SYNTAX

```
Show-InstallationProgress [[-StatusMessage] <String>] [[-WindowLocation] <String>]  
[[ -TopMost] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Create a WPF window in a separate thread to display a marquee style progress ellipse with a custom message that can be updated.

The status message supports line breaks.

The first time this function is called in a script, it will display a balloon tip notification to indicate that the installation has started (provided balloon tips are enabled in the configuration).

EXAMPLES

EXAMPLE 1

```
Show-InstallationProgress
```

Uses the default status message from the XML configuration file.

EXAMPLE 2

```
Show-InstallationProgress -StatusMessage 'Installation in Progress...'
```

EXAMPLE 3

```
Show-InstallationProgress -StatusMessage "Installation in Progress...`r`nThe  
installation may take 20 minutes to complete."
```

EXAMPLE 4

```
Show-InstallationProgress -StatusMessage 'Installation in Progress...' -  
WindowLocation 'BottomRight' -TopMost $false
```

PARAMETERS

-StatusMessage

The status message to be displayed.

The default status message is taken from the XML configuration file.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: $configProgressMessageInstall
Accept pipeline input: False
Accept wildcard characters: False
```

-TopMost

Specifies whether the progress window should be topmost.

Default: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```


-WindowLocation

The location of the progress window.

Default: center of the screen.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: Default
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-InstallationPrompt

SYNOPSIS

Displays a custom installation prompt with the toolkit branding and optional buttons.

SYNTAX

```
Show-InstallationPrompt [[-Title] <String>] [[-Message] <String>] [[-
MessageAlignment] <String>]
    [[-ButtonRightText] <String>] [[-ButtonLeftText] <String>] [[-ButtonMiddleText]
<String>] [[-Icon] <String>]
    [-NoWait] [-PersistPrompt] [[-MinimizeWindows] <Boolean>] [[-Timeout] <Int32>] [[-
ExitOnTimeout] <Boolean>]
    [[-TopMost] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Any combination of Left, Middle or Right buttons can be displayed.

The return value of the button clicked by the user is the button text specified.

EXAMPLES

EXAMPLE 1

```
Show-InstallationPrompt -Message 'Do you want to proceed with the installation?' -
ButtonRightText 'Yes' -ButtonLeftText 'No'
```

EXAMPLE 2

```
Show-InstallationPrompt -Title 'Funny Prompt' -Message 'How are you feeling today?'  
-ButtonRightText 'Good' -ButtonLeftText 'Bad' -ButtonMiddleText 'Indifferent'
```

EXAMPLE 3

```
Show-InstallationPrompt -Message 'You can customize text to appear at the end of an  
install, or remove it completely for unattended installations.' -Icon Information -  
NoWait
```

PARAMETERS

-ButtonLeftText

Show a button on the left of the prompt with the specified text

```
Type: String  
Parameter Sets: (All)  
Aliases:  
  
Required: False  
Position: 5  
Default value: None  
Accept pipeline input: False  
Accept wildcard characters: False
```

-ButtonMiddleText

Show a button in the middle of the prompt with the specified text

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ButtonRightText

Show a button on the right of the prompt with the specified text

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-ExitOnTimeout

Specifies whether to exit the script if the UI times out.

Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 10
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Icon

Show a system icon in the prompt.

Options: Application, Asterisk, Error, Exclamation, Hand, Information, None, Question, Shield, Warning, WinLogo.

Default: None

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 7
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-Message

Message text to be included in the prompt

Type: String
Parameter Sets: (All)
Aliases:

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

-MessageAlignment

Alignment of the message text.

Options: Left, Center, Right.

Default: Center.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: Center
Accept pipeline input: False
Accept wildcard characters: False

-MinimizeWindows

Specifies whether to minimize other windows when displaying prompt.

Default: \$false.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 8
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-NoWait

Specifies whether to show the prompt asynchronously (i.e. allow the script to continue without waiting for a response).
Default: \$false.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-PersistPrompt

Specify whether to make the prompt persist in the center of the screen every couple of seconds, specified in the AppDeployToolkitConfig.xml.
The user will have no option but to respond to the prompt - resistance is futile!

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-Timeout

Specifies the time period in seconds after which the prompt should timeout.
Default: UI timeout value set in the config XML file.

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 9
Default value: \$configInstallationUITimeout
Accept pipeline input: False
Accept wildcard characters: False

-Title

Title of the prompt.
Default: the application installation name.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: \$installTitle
Accept pipeline input: False
Accept wildcard characters: False

-TopMost

Specifies whether the progress window should be topmost.

Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 11
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-InstallationRestartPrompt

SYNOPSIS

Displays a restart prompt with a countdown to a forced restart.

SYNTAX

```
Show-InstallationRestartPrompt [[-CountdownSeconds] <Int32>] [[-  
CountdownNoHideSeconds] <Int32>]  
[[[-NoSilentRestart] <Boolean>] [-NoCountdown] [[-SilentCountdownSeconds] <Int32>]  
[[[-TopMost] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Displays a restart prompt with a countdown to a forced restart.

EXAMPLES

EXAMPLE 1

```
Show-InstallationRestartPrompt -Countdownseconds 600 -CountdownNoHideSeconds 60
```

EXAMPLE 2

```
Show-InstallationRestartPrompt -NoCountdown
```

EXAMPLE 3

```
Show-InstallationRestartPrompt -Countdownseconds 300 -NoSilentRestart $false -  
SilentCountdownSeconds 10
```

PARAMETERS

-CountdownNoHideSeconds

Specifies the number of seconds to display the restart prompt without allowing the window to be hidden.

Default: 30

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: 30
Accept pipeline input: False
Accept wildcard characters: False

-CountdownSeconds

Specifies the number of seconds to countdown before the system restart.

Default: 60

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: 60
Accept pipeline input: False
Accept wildcard characters: False

-NoCountdown

Specifies not to show a countdown.

The UI will restore/reposition itself persistently based on the interval value specified in the config file.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-NoSilentRestart

Specifies whether the restart should be triggered when Deploy mode is silent or very silent.

Default: \$true

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 3

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-SilentCountdownSeconds

Specifies number of seconds to countdown for the restart when the toolkit is running in silent mode and NoSilentRestart is \$false.

Default: 5

Type: Int32
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: 5
Accept pipeline input: False
Accept wildcard characters: False

-TopMost

Specifies whether the window is the topmost window.
Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.String

Returns the version of the specified file.

NOTES

Be mindful of the countdown you specify for the reboot as code directly after this function might NOT be able to execute - that includes logging.

RELATED LINKS

<https://psappdeploytoolkit.com>

Show-InstallationWelcome

SYNOPSIS

Show a welcome dialog prompting the user with information about the installation and actions to be performed before the installation can begin.

SYNTAX

None (Default)

```
Show-InstallationWelcome [-CloseApps <String>] [-Silent] [-CloseAppsCountdown
<Int32>]
[-ForceCloseAppsCountdown <Int32>] [-PromptToSave] [-PersistPrompt] [-
BlockExecution] [-AllowDefer]
[-AllowDeferCloseApps] [-DeferTimes <Int32>] [-DeferDays <Int32>] [-DeferDeadline
<String>]
[-MinimizeWindows <Boolean>] [-TopMost <Boolean>] [-ForceCountdown <Int32>] [-
CustomText] [<CommonParameters>]
```

CheckDiskSpaceParameterSet

```
Show-InstallationWelcome [-CloseApps <String>] [-Silent] [-CloseAppsCountdown
<Int32>]
[-ForceCloseAppsCountdown <Int32>] [-PromptToSave] [-PersistPrompt] [-
BlockExecution] [-AllowDefer]
[-AllowDeferCloseApps] [-DeferTimes <Int32>] [-DeferDays <Int32>] [-DeferDeadline
<String>] [-CheckDiskSpace]
[-RequiredDiskSpace <Int32>] [-MinimizeWindows <Boolean>] [-TopMost <Boolean>] [-
ForceCountdown <Int32>]
[-CustomText] [<CommonParameters>]
```

DESCRIPTION

The following prompts can be included in the welcome dialog:

- a) Close the specified running applications, or optionally close the applications without showing a prompt (using the -Silent switch).
- b) Defer the installation a certain number of times, for a certain number of days or until a deadline is reached.
- c) Countdown until applications are automatically closed.
- d) Prevent users from launching the specified applications while the installation is in progress.

Notes:

The process descriptions are retrieved from WMI, with a fall back on the process name if no description is available.

Alternatively, you can specify the description yourself with a '=' symbol - see examples.

The dialog box will timeout after the timeout specified in the XML configuration file (default 1 hour and 55 minutes) to prevent SCCM installations from timing out and returning a failure code to SCCM.

When the dialog times out, the script will exit and return a 1618 code (SCCM fast retry code).

EXAMPLES

EXAMPLE 1

```
Show-InstallationWelcome -CloseApps 'iexplore,winword,excel'
```

Prompt the user to close Internet Explorer, Word and Excel.

EXAMPLE 2

```
Show-InstallationWelcome -CloseApps 'winword,excel' -Silent
```

Close Word and Excel without prompting the user.

EXAMPLE 3

```
Show-InstallationWelcome -CloseApps 'winword,excel' -BlockExecution
```

Close Word and Excel and prevent the user from launching the applications while the installation is in progress.

EXAMPLE 4

```
Show-InstallationWelcome -CloseApps 'winword=Microsoft Office Word,excel=Microsoft Office Excel' -CloseAppsCountdown 600
```

Prompt the user to close Word and Excel, with customized descriptions for the applications and automatically close the applications after 10 minutes.

EXAMPLE 5

```
Show-InstallationWelcome -CloseApps 'winword,msaccess,excel' -PersistPrompt
```

Prompt the user to close Word, MSAccess and Excel.

By using the PersistPrompt switch, the dialog will return to the center of the screen every couple of seconds, specified in the AppDeployToolkitConfig.xml, so the user cannot ignore it by dragging it aside.

EXAMPLE 6

```
Show-InstallationWelcome -AllowDefer -DeferDeadline '25/08/2013'
```

Allow the user to defer the installation until the deadline is reached.

EXAMPLE 7

```
Show-InstallationWelcome -CloseApps 'winword,excel' -BlockExecution -AllowDefer -DeferTimes 10 -DeferDeadline '25/08/2013' -CloseAppsCountdown 600
```

Close Word and Excel and prevent the user from launching the applications while the installation is in progress.

Allow the user to defer the installation a maximum of 10 times or until the deadline is reached, whichever happens first.

When deferral expires, prompt the user to close the applications and automatically close them after 10 minutes.

PARAMETERS

-AllowDefer

Enables an optional defer button to allow the user to defer the installation.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-AllowDeferCloseApps

Enables an optional defer button to allow the user to defer the installation only if there are running applications that need to be closed.

This parameter automatically enables -AllowDefer

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-BlockExecution

Option to prevent the user from launching processes/applications, specified in - CloseApps, during the installation.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

-CheckDiskSpace

Specify whether to check if there is enough disk space for the installation to proceed.

If this parameter is specified without the RequiredDiskSpace parameter, the required disk space is calculated automatically based on the size of the script source and associated files.

```
Type: SwitchParameter
Parameter Sets: CheckDiskSpaceParameterSet
Aliases:

Required: True
Position: Named
Default value: False
Accept pipeline input: False
Accept wildcard characters: False
```

-CloseApps

Name of the process to stop (do not include the .exe).

Specify multiple processes separated by a comma.

Specify custom descriptions like this: "winword=Microsoft Office Word,excel=Microsoft Office Excel"

```
Type: String
Parameter Sets: (All)
Aliases:

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

-CloseAppsCountdown

Option to provide a countdown in seconds until the specified applications are automatically closed.

This only takes effect if deferral is not allowed or has expired.

Type: Int32
Parameter Sets: (All)
Aliases:

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

-CustomText

Specify whether to display a custom message specified in the XML file.
Custom message must be populated for each language section in the XML.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-DeferDays

Specify the number of days since first run that the installation can be deferred.
This is converted to a deadline.

Type: Int32
Parameter Sets: (All)
Aliases:

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

-DeferDeadline

Specify the deadline date until which the installation can be deferred.

Specify the date in the local culture if the script is intended for that same culture.

If the script is intended to run on EN-US machines, specify the date in the format:
"08/25/2013" or "08-25-2013" or "08-25-2013 18:00:00"

If the script is intended for multiple cultures, specify the date in the universal sortable date/time format: "2013-08-22 11:51:52Z"

The deadline date will be displayed to the user in the format of their culture.

Type: String
Parameter Sets: (All)
Aliases:

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

-DeferTimes

Specify the number of times the installation can be deferred.

```
Type: Int32
Parameter Sets: (All)
Aliases:

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

-ForceCloseAppsCountdown

Option to provide a countdown in seconds until the specified applications are automatically closed regardless of whether deferral is allowed.

```
Type: Int32
Parameter Sets: (All)
Aliases:

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

-ForceCountdown

Specify a countdown to display before automatically proceeding with the installation when a deferral is enabled.

Type: Int32
Parameter Sets: (All)
Aliases:

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

-MinimizeWindows

Specifies whether to minimize other windows when displaying prompt.
Default: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

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

-PersistPrompt

Specify whether to make the Show-InstallationWelcome prompt persist in the center of the screen every couple of seconds, specified in the AppDeployToolkitConfig.xml.
The user will have no option but to respond to the prompt.
This only takes effect if deferral is not allowed or has expired.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-PromptToSave

Specify whether to prompt to save working documents when the user chooses to close applications by selecting the "Close Programs" button.

Option does not work in SYSTEM context unless toolkit launched with "psexec.exe -s -i" to run it as an interactive process under the SYSTEM account.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-RequiredDiskSpace

Specify required disk space in MB, used in combination with CheckDiskSpace.

Type: Int32
Parameter Sets: CheckDiskSpaceParameterSet
Aliases:

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

-Silent

Stop processes without prompting the user.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-TopMost

Specifies whether the windows is the topmost window.
Default: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Start-ServiceAndDependencies

SYNOPSIS

Start Windows service and its dependencies.

SYNTAX

```
Start-ServiceAndDependencies [-Name] <String> [[-ComputerName] <String>] [-SkipServiceExistsTest]
    [-SkipDependentServices] [[-PendingStatusWait] <TimeSpan>] [-PassThru] [[-ContinueOnError] <Boolean>]
    [<CommonParameters>]
```

DESCRIPTION

Start Windows service and its dependencies.

EXAMPLES

EXAMPLE 1

```
Start-ServiceAndDependencies -Name 'wuauserv'
```

PARAMETERS

-ComputerName

Specify the name of the computer.
Default is: the local computer.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: \$env:ComputerName
Accept pipeline input: False
Accept wildcard characters: False

-ContinueOnError

Continue if an error is encountered.
Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Name

Specify the name of the service.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-PassThru

Return the System.ServiceProcess.ServiceController service object.

```
Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-PendingStatusWait

The amount of time to wait for a service to get out of a pending state before continuing.
Default is 60 seconds.

Type: TimeSpan
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: (New-TimeSpan -Seconds 60)
Accept pipeline input: False
Accept wildcard characters: False

-SkipDependentServices

Choose to skip checking for and starting dependent services.
Default is: \$false.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-SkipServiceExistsTest

Choose to skip the test to check whether or not the service exists if it was already done outside of this function.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.ServiceProcess.ServiceController.

Returns the service object.

RELATED LINKS

<https://psappdeploytoolkit.com>

Stop-ServiceAndDependencies

SYNOPSIS

Stop Windows service and its dependencies.

SYNTAX

```
Stop-ServiceAndDependencies [-Name] <String> [[-ComputerName] <String>] [-SkipServiceExistsTest]
[-SkipDependentServices] [[-PendingStatusWait] <TimeSpan>] [-PassThru] [[-ContinueOnError] <Boolean>]
[<CommonParameters>]
```

DESCRIPTION

Stop Windows service and its dependencies.

EXAMPLES

EXAMPLE 1

```
Stop-ServiceAndDependencies -Name 'wuauserv'
```

PARAMETERS

-ComputerName

Specify the name of the computer.
Default is: the local computer.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: \$env:ComputerName
Accept pipeline input: False
Accept wildcard characters: False

-ContinueOnError

Continue if an error is encountered.
Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-Name

Specify the name of the service.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-PassThru

Return the System.ServiceProcess.ServiceController service object.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-PendingStatusWait

The amount of time to wait for a service to get out of a pending state before continuing.
Default is 60 seconds.

Type: TimeSpan
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: (New-TimeSpan -Seconds 60)
Accept pipeline input: False
Accept wildcard characters: False

-SkipDependentServices

Choose to skip checking for and stopping dependent services.
Default is: \$false.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

-SkipServiceExistsTest

Choose to skip the test to check whether or not the service exists if it was already done outside of this function.

Type: SwitchParameter

Parameter Sets: (All)

Aliases:

Required: False

Position: Named

Default value: False

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.ServiceProcess.ServiceController.

Returns the service object.

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-Battery

SYNOPSIS

Tests whether the local machine is running on AC power or not.

SYNTAX

```
Test-Battery [-PassThru] [<CommonParameters>]
```

DESCRIPTION

Tests whether the local machine is running on AC power and returns true/false. For detailed information, use -PassThru option.

EXAMPLES

EXAMPLE 1

```
Test-Battery
```

EXAMPLE 2

```
(Test-Battery -PassThru).IsLaptop
```

Determines if the current system is a laptop or not.

PARAMETERS

-PassThru

Outputs a hashtable containing the following properties:

IsLaptop, IsUsingACPower, ACPowerLineStatus, BatteryChargeStatus, BatteryLifePercent, BatteryLifeRemaining, BatteryFullLifetime

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Hashtable.

Returns a hashtable containing the following properties

- IsLaptop

- IsUsingACPower
- ACPowerLineStatus
- BatteryChargeStatus
- BatteryLifePercent
- BatteryLifeRemaining
- BatteryFullLifetime

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-MSUpdates

SYNOPSIS

Test whether a Microsoft Windows update is installed.

SYNTAX

```
Test-MSUpdates [-KBNumber] <String> [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Test whether a Microsoft Windows update is installed.

EXAMPLES

EXAMPLE 1

```
Test-MSUpdates -KBNumber 'KB2549864'
```

PARAMETERS

-ContinueOnError

Suppress writing log message to console on failure to write message to log file.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 2

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

-KBNumber

KBNumber of the update.

Type: String

Parameter Sets: (All)

Aliases:

Required: True

Position: 1

Default value: None

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Boolean

Returns \$true if the update is installed, otherwise returns \$false.

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-NetworkConnection

SYNOPSIS

Tests for an active local network connection, excluding wireless and virtual network adapters.

SYNTAX

```
Test-NetworkConnection [<CommonParameters>]
```

DESCRIPTION

Tests for an active local network connection, excluding wireless and virtual network adapters, by querying the Win32_NetworkAdapter WMI class.

EXAMPLES

EXAMPLE 1

```
Test-NetworkConnection
```

PARAMETERS

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Boolean

Returns \$true if a wired network connection is detected, otherwise returns \$false.

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-PowerPoint

SYNOPSIS

Tests whether PowerPoint is running in either fullscreen slideshow mode or presentation mode.

SYNTAX

```
Test-PowerPoint [<CommonParameters>]
```

DESCRIPTION

Tests whether someone is presenting using PowerPoint in either fullscreen slideshow mode or presentation mode.

EXAMPLES

EXAMPLE 1

```
Test-PowerPoint
```

PARAMETERS

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

System.Boolean

Returns \$true if PowerPoint is running in either fullscreen slideshow mode or presentation mode, otherwise returns \$false.

NOTES

This function can only execute detection logic if the process is in interactive mode.

There is a possibility of a false positive if the PowerPoint filename starts with "PowerPoint Slide Show".

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-RegistryValue

SYNOPSIS

Test if a registry value exists.

SYNTAX

```
Test-RegistryValue [-Key] <Object> [-Value] <Object> [[-SID] <String>]  
[<CommonParameters>]
```

DESCRIPTION

Checks a registry key path to see if it has a value with a given name.
Can correctly handle cases where a value simply has an empty or null value.

EXAMPLES

EXAMPLE 1

```
Test-RegistryValue -Key 'HKLM:SYSTEM\CurrentControlSet\Control\Session Manager' -  
Value 'PendingFileRenameOperations'
```

PARAMETERS

-Key

Path of the registry key.

```
Type: Object  
Parameter Sets: (All)  
Aliases:  
  
Required: True  
Position: 1  
Default value: None  
Accept pipeline input: True (ByPropertyName, ByValue)  
Accept wildcard characters: False
```

-SID

The security identifier (SID) for a user.
Specifying this parameter will convert a HKEY_CURRENT_USER registry key to the HKEY_USERS\$SID format.

Specify this parameter from the Invoke-HKCURegistrySettingsForAllUsers function to read/edit HKCU registry settings for all users on the system.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: None
Accept pipeline input: False
Accept wildcard characters: False
```

-Value

Specify the registry key value to check the existence of.

```
Type: Object
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

System.String

Accepts a string value for the registry key path.

OUTPUTS

System.String

Returns \$true if the registry value exists, \$false if it does not.

NOTES

To test if registry key exists, use Test-Path function like so:

```
Test-Path -Path $Key -PathType 'Container'
```

RELATED LINKS

<https://psappdeploytoolkit.com>

Test-ServiceExists

SYNOPSIS

Check to see if a service exists.

SYNTAX

```
Test-ServiceExists [-Name] <String> [[-ComputerName] <String>] [-PassThru] [[-ContinueOnError] <Boolean>]  
[<CommonParameters>]
```

DESCRIPTION

Check to see if a service exists (using WMI method because Get-Service will generate ErrorRecord if service doesn't exist).

EXAMPLES

EXAMPLE 1

```
Test-ServiceExists -Name 'wuauserv'
```

EXAMPLE 2

```
Test-ServiceExists -Name 'testservice' -PassThru | Where-Object { $_ } | ForEach-Object { $_.Delete() }
```

Check if a service exists and then delete it by using the -PassThru parameter.

PARAMETERS

-ComputerName

Specify the name of the computer.

Default is: the local computer.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: $env:ComputerName
Accept pipeline input: False
Accept wildcard characters: False
```

-ContinueOnError

Continue if an error is encountered.
Default is: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

-Name

Specify the name of the service.

Note: Service name can be found by executing "Get-Service | Format-Table -AutoSize - Wrap" or by using the properties screen of a service in services.msc.

Type: String
Parameter Sets: (All)
Aliases:

Required: True
Position: 1
Default value: None
Accept pipeline input: False
Accept wildcard characters: False

-PassThru

Return the WMI service object.

To see all the properties use: Test-ServiceExists -Name 'spooler' -PassThru | Get-Member

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Update-Desktop

SYNOPSIS

Refresh the Windows Explorer Shell, which causes the desktop icons and the environment variables to be reloaded.

SYNTAX

```
Update-Desktop [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Refresh the Windows Explorer Shell, which causes the desktop icons and the environment variables to be reloaded.

EXAMPLES

EXAMPLE 1

```
Update-Desktop
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

```
Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: True
Accept pipeline input: False
Accept wildcard characters: False
```

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None. This function does not return objects.

NOTES

This function has an alias: Refresh-Desktop

RELATED LINKS

<https://psappdeploytoolkit.com>

Update-GroupPolicy

SYNOPSIS

Performs a gpupdate command to refresh Group Policies on the local machine.

SYNTAX

```
Update-GroupPolicy [[-ContinueOnError] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Performs a gpupdate command to refresh Group Policies on the local machine.

EXAMPLES

EXAMPLE 1

```
Update-GroupPolicy
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 1

Default value: True

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None

This function does not return any objects.

RELATED LINKS

<https://psappdeploytoolkit.com>

Update-SessionEnvironmentVariables

SYNOPSIS

Updates the environment variables for the current PowerShell session with any environment variable changes that may have occurred during script execution.

SYNTAX

```
Update-SessionEnvironmentVariables [-LoadLoggedOnUserEnvironmentVariables] [[-ContinueOnError] <Boolean>] [  
  <CommonParameters>]
```

DESCRIPTION

Environment variable changes that take place during script execution are not visible to the current PowerShell session.

Use this function to refresh the current PowerShell session with all environment variable settings.

EXAMPLES

EXAMPLE 1

```
Update-SessionEnvironmentVariables
```

PARAMETERS

-ContinueOnError

Continue if an error is encountered.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 1
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-LoadLoggedOnUserEnvironmentVariables

If script is running in SYSTEM context, this option allows loading environment variables from the active console user.

If no console user exists but users are logged in, such as on terminal servers, then the first logged-in non-console user.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

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

CommonParameters

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

INPUTS

None

You cannot pipe objects to this function.

OUTPUTS

None. This function does not return objects.

NOTES

This function has an alias: Refresh-SessionEnvironmentVariables

RELATED LINKS

<https://psappdeploytoolkit.com>

Write-Log

SYNOPSIS

Write messages to a log file in CMTrace.exe compatible format or Legacy text file format.

SYNTAX

```
Write-Log [-Message] <String[]> [[-Severity] <Int16>] [[-Source] <String>] [[-ScriptSection] <String>]
[[[-LogType] <String>] [[-LogFileDirectory] <String>] [[-LogFileName] <String>] [[-MaxLogFileSizeMB] <Decimal>]
[[[-WriteHost] <Boolean>] [[-ContinueOnError] <Boolean>] [-PassThru] [-DebugMessage]
[[[-LogDebugMessage] <Boolean>] [<CommonParameters>]
```

DESCRIPTION

Write messages to a log file in CMTrace.exe compatible format or Legacy text file format and optionally display in the console.

EXAMPLES

EXAMPLE 1

```
Write-Log -Message "Installing patch MS15-031" -Source 'Add-Patch' -LogType
'CMTrace'
```

EXAMPLE 2

```
Write-Log -Message "Script is running on Windows 8" -Source 'Test-ValidOS' -LogType
'Legacy'
```

EXAMPLE 3

```
Write-Log -Message "Log only message" -WriteHost $false
```

PARAMETERS

-ContinueOnError

Suppress writing log message to console on failure to write message to log file.

Default is: \$true.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 10
Default value: True
Accept pipeline input: False
Accept wildcard characters: False

-DebugMessage

Specifies that the message is a debug message.

Debug messages only get logged if -LogDebugMessage is set to \$true.

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 12
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-LogDebugMessage

Debug messages only get logged if this parameter is set to \$true in the config XML file.

Type: Boolean
Parameter Sets: (All)
Aliases:

Required: False
Position: 13
Default value: \$configToolkitLogDebugMessage
Accept pipeline input: False
Accept wildcard characters: False

-LogFileDirectory

Set the directory where the log file will be saved.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 6
Default value: \$(If (\$configToolkitCompressLogs) {
 \$logTempFolder
} Else {
 \$configToolkitLogDir
})
Accept pipeline input: False
Accept wildcard characters: False

-LogFileName

Set the name of the log file.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 7
Default value: $logName
Accept pipeline input: False
Accept wildcard characters: False
```

-LogType

Choose whether to write a CMTrace.exe compatible log file or a Legacy text log file.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 5
Default value: $configToolkitLogStyle
Accept pipeline input: False
Accept wildcard characters: False
```

-MaxLogFileSizeMB

Maximum file size limit for log file in megabytes (MB).

Default is 10 MB.

Type: Decimal
Parameter Sets: (All)
Aliases:

Required: False
Position: 8
Default value: \$configToolkitLogMaxSize
Accept pipeline input: False
Accept wildcard characters: False

-Message

The message to write to the log file or output to the console.

Type: String[]
Parameter Sets: (All)
Aliases: Text

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

-PassThru

Return the message that was passed to the function

Type: SwitchParameter
Parameter Sets: (All)
Aliases:

Required: False
Position: 11
Default value: False
Accept pipeline input: False
Accept wildcard characters: False

-ScriptSection

The heading for the portion of the script that is being executed.
Default is: \$script:installPhase.

Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 4
Default value: \$script:installPhase
Accept pipeline input: False
Accept wildcard characters: False

-Severity

Defines message type.

When writing to console or CMTrace.exe log format, it allows highlighting of message type.

Options: 1 = Information (default), 2 = Warning (highlighted in yellow), 3 = Error (highlighted in red)

```
Type: Int16
Parameter Sets: (All)
Aliases:

Required: False
Position: 2
Default value: 1
Accept pipeline input: False
Accept wildcard characters: False
```

-Source

The source of the message being logged.

```
Type: String
Parameter Sets: (All)
Aliases:

Required: False
Position: 3
Default value: $([String]$parentFunctionName =
[IO.Path]::GetFileNameWithoutExtension((Get-Variable -Name 'MyInvocation' -Scope 1
-ErrorAction 'SilentlyContinue').Value.MyCommand.Name); If ($parentFunctionName) {
    $parentFunctionName
} Else {
    'Unknown'
})
Accept pipeline input: False
Accept wildcard characters: False
```

-WriteHost

Write the log message to the console.

Type: Boolean

Parameter Sets: (All)

Aliases:

Required: False

Position: 9

Default value: \$configToolkitLogWriteToHost

Accept pipeline input: False

Accept wildcard characters: False

CommonParameters

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

INPUTS

System.String

The message to write to the log file or output to the console.

OUTPUTS

None

This function does not generate any output.

RELATED LINKS

<https://psappdeploytoolkit.com>
