

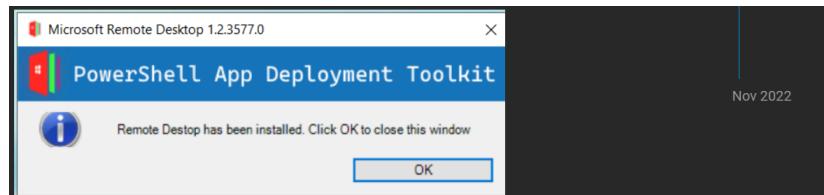
Commands For Powershell AppDeploy Toolkit.

1. Install Application:



```
379
380
381 #-----
382 # MARK: Wrapper around Get-ADTApplication
383 #
384 #-----
385 Execute-MSI -Action Install -Patch "C:\path\to\app.msi"-Parameters "/qn REBOOT =ReallySuppress
386 #Perform Installing In AppDeploy Toolkit
387
388 function Get-InstalledApplication
389 {
390     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'Name', Justification = "This parameter is passed to an underlying function via '$PSBoundaries'")]
391     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'ProductCode', Justification = "This parameter is passed to an underlying function via '$PSBoundaries'")]
392     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'IncludeUpdatesAndHotfixes', Justification = "This parameter is passed to an underlying function via '$PSBoundaries'")]
393     [CmdletBinding()]
394     param (
395         [Parameter(Mandatory = $false)]
396         [ValidateNotNullOrEmpty()]
397         [System.String[]]$Name,
398
399         [Parameter(Mandatory = $false)]
400         [ValidateNotNullOrEmpty()]
401         [System.String]$ProductCode,
402
403         [Parameter(Mandatory = $false)]
404         [System.Management.Automation.SwitchParameter]$Exact,
405
406         [Parameter(Mandatory = $false)]
407         [System.Management.Automation.SwitchParameter]$WildCard,
408
409         [Parameter(Mandatory = $false)]
410         [System.Management.Automation.SwitchParameter]$RegEx,
411
412         [Parameter(Mandatory = $false)]
413         [System.Management.Automation.SwitchParameter]$IncludeUpdatesAndHotfixes
414     )
415
416     # Set strict mode to the highest within this function's scope.
417     Set-StrictMode -Version 3
418
419     # Announce overall deprecation.
420     Write-ADTLogEntry -Message "The function '$($MyInvocation.MyCommand.Name)' has been replaced by 'Get-ADTApplication'. Please migrate your scripts to use the new function." -S
421
422 }
```

Output:



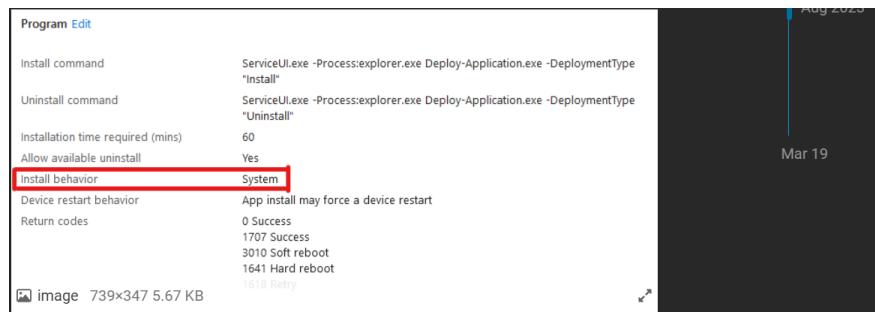
2.Uninstall Application:

```

Untitled1.ps1 AppDeployToolkitMain.ps1* X
448
449
450
451 #-----#
452 # MARK: Wrapper around Uninstall-ADTApplication
453 #
454
455 Uninstall-ADTApplication -Name 'Acrobat' -ApplicationType 'MSI' -FilterScript { $_.Publisher -match 'Adobe' }
456 #Removes all MSI applications that contain the name 'Acrobat' in the DisplayName and 'Adobe' in the Publisher name
457 function Remove-MSIApplications
458 {
459     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '')]
460     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', '$FilterScript')]
461     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', '$Name')]
462     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSAvoidAssignmentToAutomaticVariables', '$Is')]
463     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'ArgumentList')]
464     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'AdditionalArgumentList')]
465     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'LoggingOptions')]
466     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'IncludeUpdatesAndHotfixes')]
467     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'PassThru')]
468     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'Exact')]
469     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'WildCard')]
470     [CmdletBinding()]
471     param
472     (
473         [Parameter(Mandatory = $true)]
474         [ValidateNotNullOrEmpty()]
475         [System.String]$Name,
476
477         [Parameter(Mandatory = $false)]
478         [System.Management.Automation.SwitchParameter]$Exact,
479
480         [Parameter(Mandatory = $false)]
481         [System.Management.Automation.SwitchParameter]$WildCard,
482
483         [Parameter(Mandatory = $false)]
484         [Alias('Arguments')]
485         [ValidateNotNullOrEmpty()]
486         [System.String]$ArgumentList,
487
488         [Parameter(Mandatory = $false)]
489         [ValidateNotNullOrEmpty()]
490         [Alias('AddParameters')]
491     )
492 }

```

Output:



3 . Copy File:

```

Untitled1.ps1 AppDeployToolkitMain.ps1* X
796
797 #-----#
798 # MARK: Wrapper around Copy-ADTFile
799 #
800
801 Copy-ADTFile -Path 'C:\Path\file.txt' -Destination 'D:\Destination\file.txt'
802 #Copies files and directories from a source to a destination. This function supports recursive copying, overwriting existing files, and returning the copied items.
803 function Copy-File
804 {
805     [CmdletBinding(SupportsShouldProcess = $false)]
806     param
807     (
808         [Parameter(Mandatory = $true, Position = 0)]
809         [ValidateNotNullOrEmpty()]
810         [System.String[]]$Path,
811
812         [Parameter(Mandatory = $true, Position = 1)]
813         [ValidateNotNullOrEmpty()]
814         [System.String]$Destination,
815
816         [Parameters(Mandatory = $false)]
817         [System.Management.Automation.SwitchParameter]$Recurse,
818
819         [Parameter(Mandatory = $false)]
820         [System.Management.Automation.SwitchParameter]$Flatten,
821
822         [Parameter(Mandatory = $false)]
823         [ValidateNotNullOrEmpty()]
824         [System.Boolean]$ContinueOnError = $true,
825
826         [Parameter(Mandatory = $false)]
827         [ValidateNotNullOrEmpty()]
828         [System.Boolean]$ContinueFileCopyOnError = $false,
829
830         [Parameter(Mandatory = $false)]
831         [ValidateNotNullOrEmpty()]
832         [System.Boolean]$UseRobocopy,
833
834         [Parameter(Mandatory = $false)]
835         [ValidateNotNullOrEmpty()]
836         [System.String]$RobocopyParams = '/NOSH /NJS /NS /NC /NP /NDL /FP /IS /IT /IM /XX /MT:4 /R:1 /W:1',
837
838         [Parameter(Mandatory = $false)]
839         [System.String]$RobocopyAdditionalParams

```

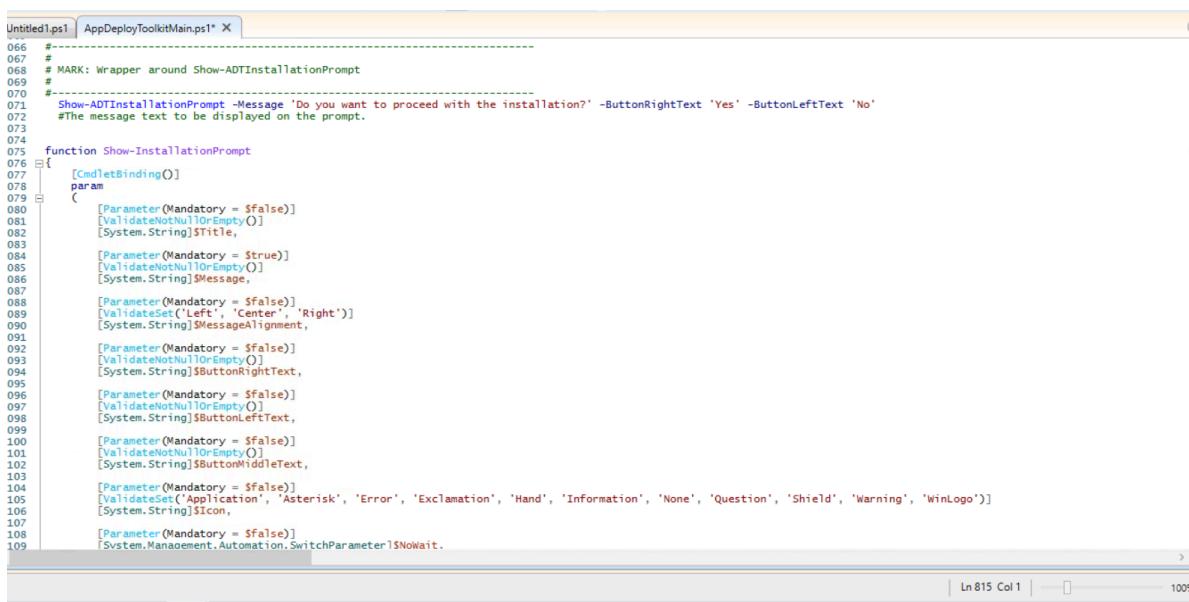
Output:

```

300 Copy-Item -Path "$dirSupportFiles\tax.exe" -Destination "C:\Temp" -Force
301
PS C:\temp\xcess> Copy-Item -Path "$dirSupportFiles\tax.exe" -Destination "C:\Temp" -Force
Copy-Item : Cannot find path 'C:\tax.exe' because it does not exist.
At line:1 char:7
+ Copy-Item -Path "$dirSupportFiles\tax.exe" -Destination "C:\Temp" ...
+ CategoryInfo          : ObjectNotFound: (C:\tax.exe:String) [Copy-Item], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.CopyItemCommand
PS C:\temp\xcess>

```

4. Installation Prompt:

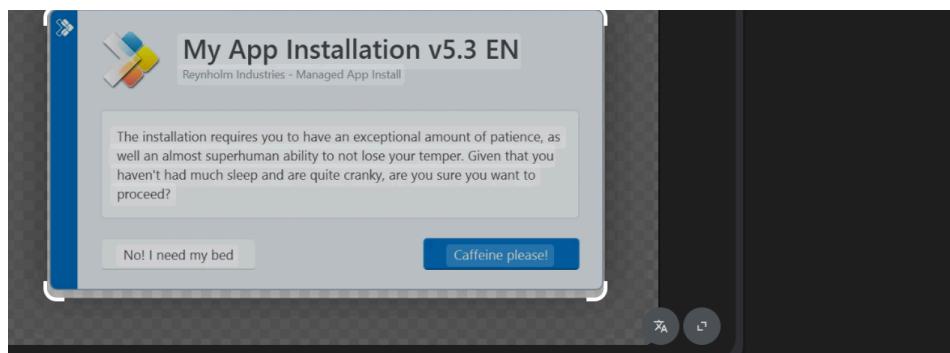


```

Untitled1.ps1 AppDeployToolkitMain.ps1 X
-----
066 #
067 #
068 # MARK: Wrapper around Show-ADTInstallationPrompt
069 #
070 #
071 Show-ADTInstallationPrompt -Message 'Do you want to proceed with the installation?' -ButtonRightText 'Yes' -ButtonLeftText 'No'
072 #The message text to be displayed on the prompt.
073
074 function Show-InstallationPrompt
075 {
076     [CmdletBinding()]
077     param
078     (
079         [Parameter(Mandatory = $false)]
080         [ValidateNotNullOrEmpty()]
081         [System.String]$Title,
082
083         [Parameter(Mandatory = $true)]
084         [ValidateNotNullOrEmpty()]
085         [System.String]$Message,
086
087         [Parameter(Mandatory = $false)]
088         [ValidateSet('Left', 'Center', 'Right')]
089         [System.String]$MessageAlignment,
090
091         [Parameter(Mandatory = $false)]
092         [ValidateNotNullOrEmpty()]
093         [System.String]$ButtonTextRightText,
094
095         [Parameter(Mandatory = $false)]
096         [ValidateNotNullOrEmpty()]
097         [System.String]$ButtonTextLeftText,
098
099         [Parameter(Mandatory = $false)]
100        [ValidateNotNullOrEmpty()]
101        [System.String]$ButtonTextMiddleText,
102
103        [Parameter(Mandatory = $false)]
104        [ValidateSet('Application', 'Asterisk', 'Error', 'Exclamation', 'Hand', 'Information', 'None', 'Question', 'Shield', 'Warning', 'WinLogo')]
105        [System.String]$Icon,
106
107        [Parameter(Mandatory = $false)]
108        [System.Management.Automation.SwitchParameter]$NoWait,
109
110    )

```

Output:



5. Installation Progress:

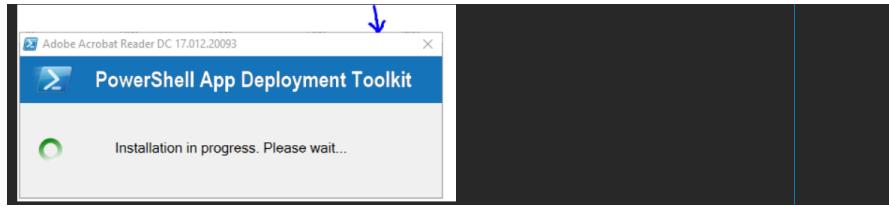
Windows PowerShell ISE

```

Untitled1.ps1 AppDeployToolkitMain.ps1* X
1165 #-----#
1166 # MARK: Wrapper around Show-ADTInstallationProgress
1167 #
1168 #-----#
1169 Show-ADTInstallationProgress
1170 #Displays a progress dialog with the status message 'Installation in Progress...'.
1171 function Show-InstallationProgress
1172 {
1173     [CmdletBinding()]
1174     param
1175     (
1176         [Parameter(Mandatory = $false)]
1177         [ValidateNotNullOrEmpty()]
1178         [System.String]$StatusMessage,
1179         [Parameter(Mandatory = $false)]
1180         [ValidateSet('Default', 'TopLeft', 'Top', 'TopRight', 'TopCenter', 'BottomLeft', 'Bottom', 'BottomRight')]
1181         [System.String]$WindowLocation,
1182         [Parameter(Mandatory = $false)]
1183         [ValidateNotNullOrEmpty()]
1184         [System.Boolean]$StopMost = $true,
1185         [Parameter(Mandatory = $false)]
1186         [System.Management.Automation.SwitchParameter]$Quiet,
1187         [Parameter(Mandatory = $false)]
1188         [System.Management.Automation.SwitchParameter]$NoRelocation
1189     )
1190     # Set strict mode to the highest within this function's scope.
1191     Set-StrictMode -Version 3
1192     # Announce overall deprecation before executing.
1193     Write-ADTLogEntry -Message "The Function [${$MyInvocation.MyCommand.Name}] has been replaced by [Show-ADTInstallationProgress]. Please migrate your scripts to use the new function."
1194     if ($PSBoundParameters.ContainsKey('TopMost'))
1195     {
1196         $PSBoundParameters.Add('NotTopMost', !$PSBoundParameters.TopMost)
1197         $null = $PSBoundParameters.Remove('TopMost')
1198     }
1199     if ($PSBoundParameters.ContainsKey('Quiet'))
1200     {
1201         $PSBoundParameters.Add('InformationAction', [System.Management.Automation.ActionPreference]::SilentlyContinue)
1202         $null = $PSBoundParameters.Remove('Quiet')
1203     }
1204     #-----#
1205     #-----#
1206     #-----#
1207     #-----#
1208     #-----#

```

Output:



6. Show DialogBox:

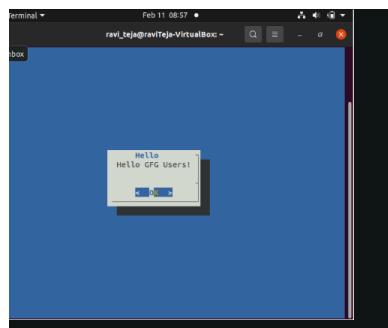
Windows PowerShell ISE

```

Untitled1.ps1 AppDeployToolkitMain.ps1* X
#-----#
# MARK: Wrapper around Show-ADTDialogBox
#-----#
Show-ADTDialogBox -Title 'Installation Notice' -Text 'Installation will take approximately 30 minutes. Do you wish to proceed?' -Buttons 'OKCancel' -DefaultButton 'OK'
#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
function Show-DialogBox
{
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true, Position = 0, HelpMessage = 'Enter a message for the dialog box.')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Text,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Title,
        [Parameter(Mandatory = $false)]
        [ValidateSet('OK', 'OKCancel', 'AbortRetryIgnore', 'YesNoCancel', 'YesNo', 'RetryCancel', 'CancelTryAgainContinue')]
        [System.String]$Buttons,
        [Parameter(Mandatory = $false)]
        [ValidateSet('First', 'Second', 'Third')]
        [System.String]$DefaultButton,
        [Parameter(Mandatory = $false)]
        [ValidateSet('Exclamation', 'Information', 'None', 'Stop', 'Question')]
        [System.String]$Icon,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Timeout,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$StopMost
    )
}

```

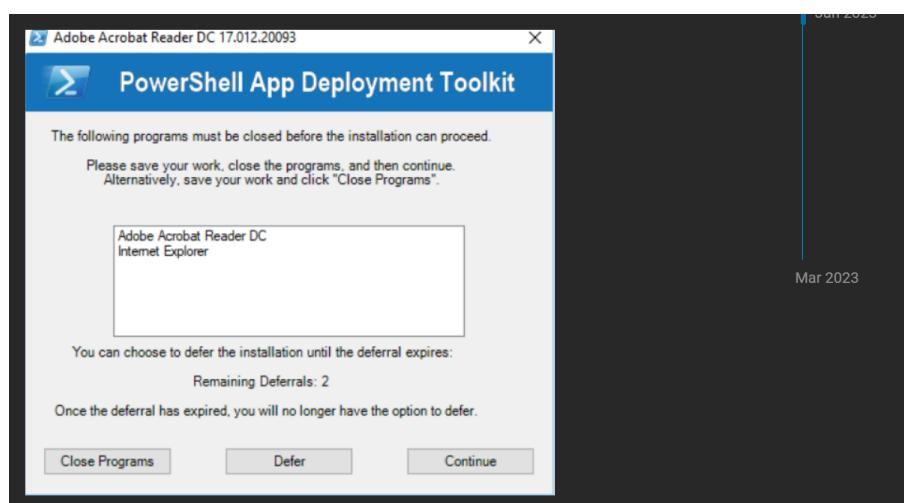
Output:



7. Show Installation Welcome:

```
titled1.ps1 AppDeployToolkitMain.ps1* 
2
3
4
5 # MARK: Wrapper around Show-ADTInstallationWelcome
6 #
7 #
8 #
9 Show-ADTInstallationWelcome -CloseProcesses iexplore, winword, excel
10 #Prompt the user to close Internet Explorer, Word and Excel.
11 function Show-InstallationWelcome
12 {
13     [CmdletBinding(DefaultParameterSetName = 'None')]
14     param
15     (
16         [Parameter(Mandatory = $false)]
17         [ValidateNotNullOrEmpty()]
18         [System.String]$CloseApps,
19
20         [Parameter(Mandatory = $false)]
21         [System.Management.Automation.SwitchParameter]$Silent,
22
23         [Parameter(Mandatory = $false)]
24         [ValidateNotNullOrEmpty()]
25         [System.Int32]$CloseAppsCountdown,
26
27         [Parameter(Mandatory = $false)]
28         [ValidateNotNullOrEmpty()]
29         [System.Int32]$ForceCloseAppsCountdown,
30
31         [Parameter(Mandatory = $false)]
32         [System.Management.Automation.SwitchParameter]$PromptToSave,
33
34         [Parameter(Mandatory = $false)]
35         [System.Management.Automation.SwitchParameter]$PersistPrompt,
36
37         [Parameter(Mandatory = $false)]
38         [System.Management.Automation.SwitchParameter]$BlockExecution,
39
40         [Parameter(Mandatory = $false)]
41         [System.Management.Automation.SwitchParameter]$AllowDefer,
42
43         [Parameter(Mandatory = $false)]
44         [System.Management.Automation.SwitchParameter]$AllowDeferCloseApps,
```

Output:

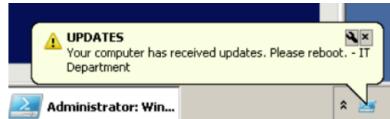


8.Show Balloon Tip:

```
hd1.ps1 AppDeployToolkitMain.ps1* X
-----
#
# MARK: Wrapper around Show-ADTBalloonTip
#
#<-- BalloonTip -BalloonTipText 'Installation Started' -BalloonTipTitle 'Application Name'
# Displays a balloon tip notification in the system tray.
function Show-BalloonTip
{
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true, Position = 0)]
        [ValidateNotNullOrEmpty()]
        [System.String]$BalloonTipText,
        [Parameter(Mandatory = $false, Position = 1)]
        [ValidateNotNullOrEmpty()]
        [System.String]$BalloonTipTitle,
        [Parameter(Mandatory = $false, Position = 2)]
        [ValidateSet('Error', 'Info', 'None', 'Warning')]
        [System.Windows.Forms.ToolTipIcon]$BalloonTipIcon,
        [Parameter(Mandatory = $false, Position = 3)]
        [ValidateNotNullOrEmpty()]
        [System.Int32]$BalloonTipTime,
        [Parameter(Mandatory = $false, Position = 4)]
        [System.Management.Automation.SwitchParameter]$NoWait
    )
    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3

    Write-ADTLogEntry -Message "The function [$(MyInvocation.MyCommand.Name)] has been replaced by [Show-ADTBalloonTip]. Please migrate your scripts to use the new function."
    if ($NoWait)
    {
        Write-ADTLogEntry -Message "The parameter '-NoWait' is discontinued and no longer has any effect." -Severity 2 -Source MyInvocation.MyCommand.Name
        $null = $PSBoundParameters.Remove('NoWait')
    }
    try
    {
        Show-ADTBalloonTip @PSBoundParameters
    }
}
```

Output:



9.Get Invalue:

```

0 #-----
1 # MARK: Wrapper around Get-ADTIniValue
2 #
3 #-----Get-ADTIniValue -FilePath "Env:\ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -Key 'KeyFileName'
4 #The Get-ADTIniValue function parses an INI file and returns the value of the specified section and key. This function is useful for retrieving configuration settings stored in
5 function Get-ADTIniValue
6 {
7     [CmdletBinding()]
8     param(
9         [Parameter(Mandatory = $true)]
10        [ValidateScript({
11            if (![[System.IO.File]::Exists($_)])
12            {
13                $PSCmdlet.ThrowTerminatingError([New-ADTValidateScriptErrorRecord -ParameterName FilePath -ProvidedValue $_ -ExceptionMessage 'The specified file does not exist'])
14            }
15            return ![[System.String]::IsNullOrEmptySpace($_)]
16        })]
17        [System.String]$filePath,
18        [Parameter(Mandatory = $true)]
19        [ValidateNotNullOrEmpty()]
20        [System.String]$section,
21        [Parameter(Mandatory = $true)]
22        [ValidateNotNullOrEmpty()]
23        [System.String]$key,
24        [Parameter(Mandatory = $false)]
25        [ValidateNotNullOrEmpty()]
26        [System.Boolean]$ContinueOnError = $true
27    )
28
29    # Set strict mode to the highest within this function's scope.
30    Set-StrictMode -Version 3
31
32    Write-ADTLogEntry -Message "The function [${$MyInvocation.MyCommand.Name}] has been replaced by [Get-ADTIniValue]. Please migrate your scripts to use the new function." -Severity Information
33    if ($PSBoundParameters.ContainsKey('ContinueOnError'))
34    {
35
36

```

10. Set IniValue:

```

1 #-----
2 # MARK: Wrapper around Set-ADTIniValue
3 #
4 #-----Set-ADTIniValue -FilePath "Env:\ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -Key 'KeyFileName' -Value 'MyFile.ID'
5 #Sets the 'KeyFileName' key in the 'Notes' section of the 'notes.ini' file to 'MyFile.ID'.
6 function Set-ADTIniValue
7 {
8     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot support ShouldProcess")]
9     [CmdletBinding(SupportsShouldProcess = $false)]
10    param(
11        [Parameter(Mandatory = $true)]
12        [ValidateScript({
13            if (![[System.IO.File]::Exists($_)])
14            {
15                $PSCmdlet.ThrowTerminatingError([New-ADTValidateScriptErrorRecord -ParameterName FilePath -ProvidedValue $_ -ExceptionMessage 'The specified file does not exist'])
16            }
17            return ![[System.String]::IsNullOrEmptySpace($_)]
18        })]
19        [System.String]$filePath,
20        [Parameter(Mandatory = $true)]
21        [ValidateNotNullOrEmpty()]
22        [System.String]$section,
23        [Parameter(Mandatory = $true)]
24        [ValidateNotNullOrEmpty()]
25        [System.String]$key,
26        [Parameter(Mandatory = $true)]
27        [AllowNull()]
28        [System.Object]$value,
29        [Parameter(Mandatory = $false)]
30        [ValidateNotNullOrEmpty()]
31        [System.Boolean]$ContinueOnError = $true
32    )
33
34    # Set strict mode to the highest within this function's scope.
35    Set-StrictMode -Version 3
36

```

11. Start services and Dependencies:

```

1881 #
1882 #-----#
1883 #
1884 # MARK: Wrapper around Start-ADTServiceAndDependencies
1885 #
1886 #
1887 Start-ADTServiceAndDependencies -Name 'wuauserv'
1888 #Starts the Windows Update service and its dependencies.
1889 function Start-ServiceAndDependencies
1890 {
1891     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot support this feature")]
1892     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for backward compatibility")]
1893     [CmdletBinding()]
1894     param
1895     (
1896         [Parameter(Mandatory = $true)]
1897         [ValidateNotNullOrEmpty()]
1898         [Alias('Name')]
1899         [System.String]$Service,
1900
1901         [Parameter(Mandatory = $false)]
1902         [ValidateNotNullOrEmpty()]
1903         [System.String]$ComputerName,
1904
1905         [Parameter(Mandatory = $false)]
1906         [System.Management.Automation.SwitchParameter]$SkipServiceExistsTest,
1907
1908         [Parameter(Mandatory = $false)]
1909         [System.Management.Automation.SwitchParameter]$SkipDependentServices,
1910
1911         [Parameter(Mandatory = $false)]
1912         [ValidateNotNullOrEmpty()]
1913         [System.TimeSpan]$PendingStatusWait,
1914
1915         [Parameter(Mandatory = $false)]
1916         [System.Management.Automation.SwitchParameter]$PassThru,
1917
1918         [Parameter(Mandatory = $false)]
1919         [ValidateNotNullOrEmpty()]
1920         [System.Boolean]$ContinueOnError = $true
1921
1922     )
1923
1924     # Set strict mode to the highest within this function's scope.
1925     Set-StrictMode -Version 3

```

12.Stop Services and Dependencies:

```

2959 #
2960 #-----#
2961 #
2962 # MARK: Wrapper around Stop-ADTServiceAndDependencies
2963 #
2964 #
2965 Stop-ADTServiceAndDependencies -Name 'wuauserv'
2966 #Stops the Windows Update service and its dependencies.
2967 function Stop-ServiceAndDependencies
2968 {
2969     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot support this feature")]
2970     [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for backward compatibility")]
2971     [CmdletBinding()]
2972     param
2973     (
2974         [Parameter(Mandatory = $true)]
2975         [ValidateNotNullOrEmpty()]
2976         [Alias('Name')]
2977         [System.String]$Service,
2978
2979         [Parameter(Mandatory = $false)]
2980         [ValidateNotNullOrEmpty()]
2981         [System.String]$ComputerName,
2982
2983         [Parameter(Mandatory = $false)]
2984         [System.Management.Automation.SwitchParameter]$SkipServiceExistsTest,
2985
2986         [Parameter(Mandatory = $false)]
2987         [System.Management.Automation.SwitchParameter]$SkipDependentServices,
2988
2989         [Parameter(Mandatory = $false)]
2990         [ValidateNotNullOrEmpty()]
2991         [System.TimeSpan]$PendingStatusWait,
2992
2993         [Parameter(Mandatory = $false)]
2994         [System.Management.Automation.SwitchParameter]$PassThru,
2995
2996         [Parameter(Mandatory = $false)]
2997         [ValidateNotNullOrEmpty()]
2998         [System.Boolean]$ContinueOnError = $true
2999
3000
3001     # Set strict mode to the highest within this function's scope.
3002     Set-StrictMode -Version 3
3003

```

13.Set Registry Key:

```

#-
# MARK: Wrapper around Set-ADTRegistryKey
#
#-----#
Set-ADTRegistryKey -Key $BlockedAppPath -Name 'Debugger' -Value $BlockedAppDebuggerValue
#Creates or sets the 'Debugger' value in the specified registry key.
function Set-RegistryKey
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot sup
    [CmdletBinding(SupportsShouldProcess = $False)]
    param
    (
        [Parameter(Mandatory = $True)]
        [ValidateNotNullOrEmpty()]
        [System.String]$key,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.String]$name,
        [Parameter(Mandatory = $False)]
        [System.Object]$value,
        [Parameter(Mandatory = $False)]
        [ValidateSet('Binary', 'Dword', 'ExpandString', 'None', 'QWord', 'String', 'Unknown')]
        [Microsoft.Win32.RegistryValueKind]$type,
        [Parameter(Mandatory = $False)]
        [System.Management.Automation.SwitchParameter]$Wow6432Node,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.String]$SID,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $True
    )
    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
}

```

14.Remove Registry Key:

```

#-
# MARK: Wrapper around Remove-ADTRegistryKey
#
#-----#
Remove-ADTRegistryKey -Key 'HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce'
#Deleted the specified registry key.

function Remove-RegistryKey
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot s
    [CmdletBinding(SupportsShouldProcess = $False)]
    param
    (
        [Parameter(Mandatory = $True)]
        [ValidateNotNullOrEmpty()]
        [System.String]$key,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.String]$name,
        [Parameter(Mandatory = $False)]
        [System.Management.Automation.SwitchParameter]$Recurse,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.String]$SID,
        [Parameter(Mandatory = $False)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $True
    )
    # Set strict mode to the highest within this Function's scope.
    Set-StrictMode -Version 3
    # Announce overall deprecation and translate $ContinueOnError to an ActionPreference before executing.
    Write-ADTLogEntry -Message "The function [${$MyInvocation.MyCommand.Name}] has been replaced by [Remove-ADTRegistryKey]. Please migrate your scripts to use the new functi
    if ($PSBoundParameters.ContainsKey('ContinueOnError'))
    {
        $null = $PSBoundParameters.Remove('ContinueOnError')
    }
}

```

15.Get Registry Key:

```

#-----
# MARK: Wrapper around Get-ADTRegistryKey
#
#-----Get-ADTRegistryKey -Key 'HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{1AD147D0-BE0E-3D6C-AC11-64F6DC4163F1'
#This example retrieves all value names and data for the specified registry key.

function Get-RegistryKey
{
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$key,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [Alias('Value')]
        [System.String]$Name,
        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Wow6432Node,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$SID,
        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$ReturnEmptyKeyIfExists,
        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$DoNotExpandEnvironmentNames,
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true
    )
    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
}

```

16. Install MSUpdates:

```

[]

#-----
# MARK: Wrapper around Install-ADTMSUpdates
#
#-----Install-ADTMSUpdates -Directory "$(Get-SessionDirFiles)\MSUpdates"
#Installs all Microsoft Updates found in the specified directory.
function Install-MSUpdates
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for legacy compatibility")]
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Directory
    )
    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
    Write-ADTLogEntry -Message "The function [$(MyInvocation.MyCommand.Name)] has been replaced by [Install-ADTMSUpdates]. Please migrate your scripts to use the new function"
    try
    {
        Install-ADTMSUpdates @PSBoundParameters
    }
    catch
    {
        $PSCmdlet.ThrowTerminatingError($_)
    }
}

```

17. Set Active Setup:

```

#-----
# MARK: Wrapper around Set-ADTActiveSetup
#
#-----Set-ADTActiveSetup -StubExePath 'C:\Users\Public\Company\ProgramUserConfig.vbs' -Arguments '/Silent' -Description 'Program User Config' -Key 'ProgramUserConfig' -Locale 'en'
#Active Setup allows handling of per-user changes registry/file changes upon login.

function Set-ActiveSetup
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot be removed without breaking compatibility with existing scripts")]
    [CmdletBinding(SupportsShouldProcess = $false)]
    param
    (
        [Parameter(Mandatory = $true, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$StubExePath,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Arguments,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Description,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Key,
        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Wow6432Node,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Version,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
        [ValidateNotNullOrEmpty()]
        [System.String]$Locale,
        [Parameter(Mandatory = $false, ParameterSetName = 'Create')]
    )
}

```

18.Installation Progress:

```
#-----  
# MARK: Wrapper around Close-ADTInstallationProgress  
#-----  
Show-ADTInstallationProgress  
#uses the default status message from the strings.ps1 file.  
function Close-ADTInstallationProgress  
{  
    [CmdletBinding()  
    param  
    (  
        [Parameter(Mandatory = $false)]  
        [ValidateRange(1, 60)]  
        [System.Int32]$WaitingTime = 5  
    )  
    # Set strict mode to the highest within this function's scope.  
    Set-StrictMode -Version 3  
    # Announce overall deprecation and any dead parameters before executing.  
    Write-ADTLogEntry -Message "The Function [${$MyInvocation.MyCommand.Name}] has been replaced by [Close-ADTInstallationProgress]. Please migrate your scripts to use the new  
if ($PSBoundParameters.ContainsKey('WaitingTime'))  
{  
    Write-ADTLogEntry -Message "The parameter '-WaitingTime' is discontinued and no longer has any effect." -Severity 2 -Source $MyInvocation.MyCommand.Name  
}  
# Invoke underlying function.  
try  
{  
    Close-ADTInstallationProgress  
}  
catch  
{  
    $PSCmdlet.ThrowTerminatingError($_)  
}
```