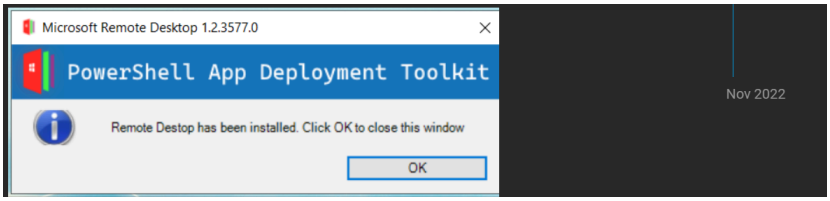
Commands For Powershell AppDeploy Toolkit.

1. Install Application:

```
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Get-ADTApplication
#
#-----------------------------------------------------------------------------
Execute-MSI -Action Install -Patch "C:\path\to\app.msi-Parameters"/qn REBOOT =Reallysuppress
#Perform Installing In AppDeploy Toolkit

function Get-InstalledApplication
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'Name', Justification = "This parameter is passed to an underlying function via `$PSBound
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'ProductCode', Justification = "This parameter is passed to an underlying function via `$
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'IncludeUpdatesAndHotfixes', Justification = "This parameter is passed to an underlying f
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String[]]$Name,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$ProductCode,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Exact,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$WildCard,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$RegEx,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$IncludeUpdatesAndHotfixes
    )

    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3

    # Announce overall deprecation.
    Write-ADTLogEntry -Message "The function [$($MyInvocation.MyCommand.Name)] has been replaced by [Get-ADTApplication]. Please migrate your scripts to use the new function." -S
```

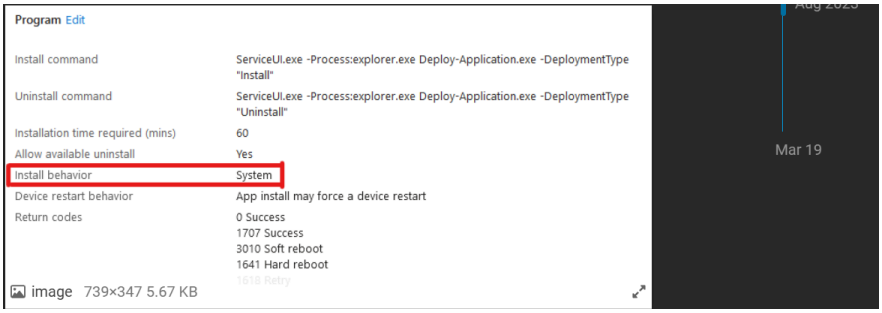
Output:



2.Uninstall Application:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Uninstall-ADTApplication
#
#-----------------------------------------------------------------------------
Uninstall-ADTApplication -Name 'Acrobat' -ApplicationType 'MSI' -FilterScript { $_.Publisher -match 'Adobe' }
#Removes all MSI applications that contain the name 'Acrobat' in the DisplayName and 'Adobe' in the Publisher name
function Remove-MSIApplications
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot suppo
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for back
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'Name', Justification = "This parameter is passed to an underlying function via '$PSBounc
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSAvoidAssignmentToAutomaticVariable', '', Justification = '$_ is intentionally overwritten in this function to exp
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'ArgumentList', Justification = "This parameter is passed to an underlying function via '
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'AdditionalArgumentList', Justification = "This parameter is passed to an underlying func
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'IncludeUpdatesAndHotfixes', Justification = "This parameter is passed to an underlying f
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'LoggingOptions', Justification = "This parameter is passed to an underlying function via
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'LogFileName', Justification = "This parameter is passed to an underlying function via '$
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'PassThru', Justification = "This parameter is passed to an underlying function via '$PSE
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'Exact', Justification = "This parameter is used within delegates that PSScriptAnalyzer h
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSReviewUnusedParameter', 'WildCard', Justification = "This parameter is used within delegates that PSScriptAnalyze
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Name,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Exact,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$WildCard,

        [Parameter(Mandatory = $false)]
        [Alias('Arguments', 'Parameters')]
        [ValidateNotNullOrEmpty()]
        [System.String]$ArgumentList,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [Alias('AddParameters')]
```

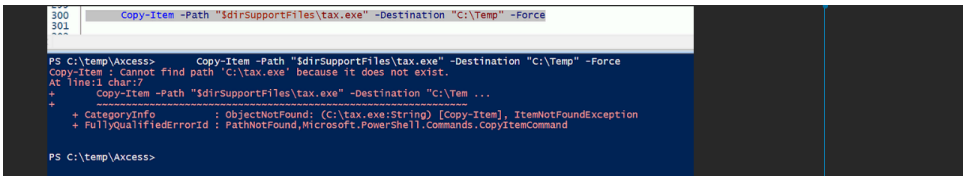
Output:



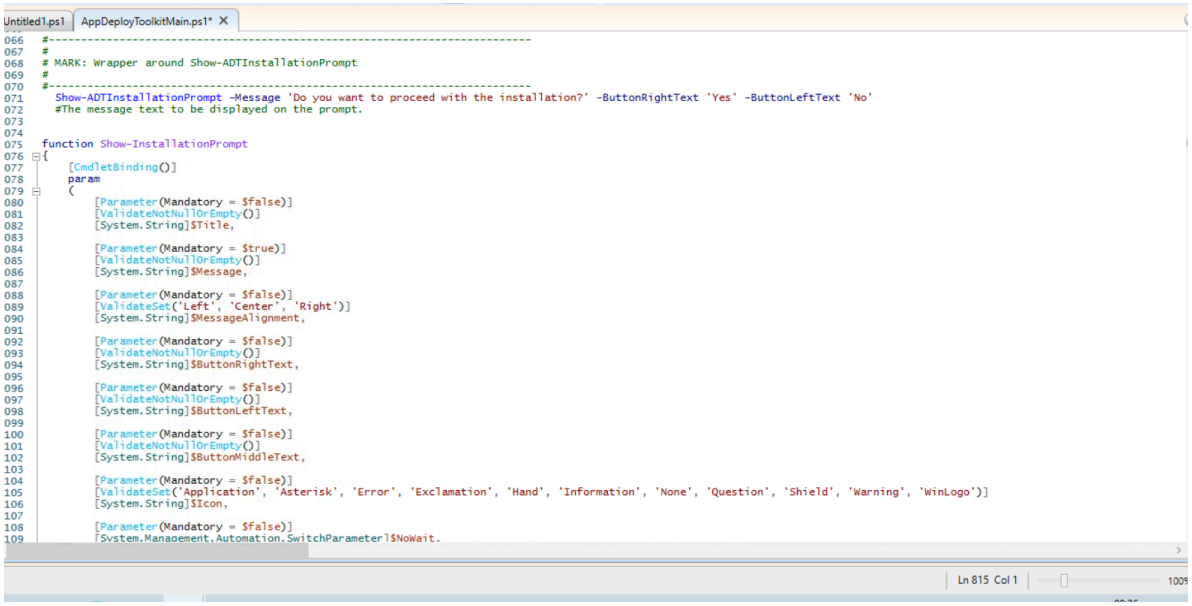
3 . Copy File:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Copy-ADTFile
#
#-----------------------------------------------------------------------------
Copy-ADTFile -Path 'C:\Path\file.txt' -Destination 'D:\Destination\file.txt'
#Copies files and directories from a source to a destination. This function supports recursive copying, overwriting existing files, and returning the copied items.
function Copy-File
{
    [CmdletBinding(SupportsShouldProcess = $false)]
    param
    (
        [Parameter(Mandatory = $true, Position = 0)]
        [ValidateNotNullOrEmpty()]
        [System.String[]]$Path,

        [Parameter(Mandatory = $true, Position = 1)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Destination,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Recurse,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Flatten,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueFileCopyOnError = $false,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$UseRobocopy,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$RobocopyParams = '/NJH /NJS /NS /NC /NP /NDL /FP /IS /IT /IM /XX /MT:4 /R:1 /W:1',

        [Parameter(Mandatory = $false)]
        [System.String]$RobocopyAdditionalParams
```

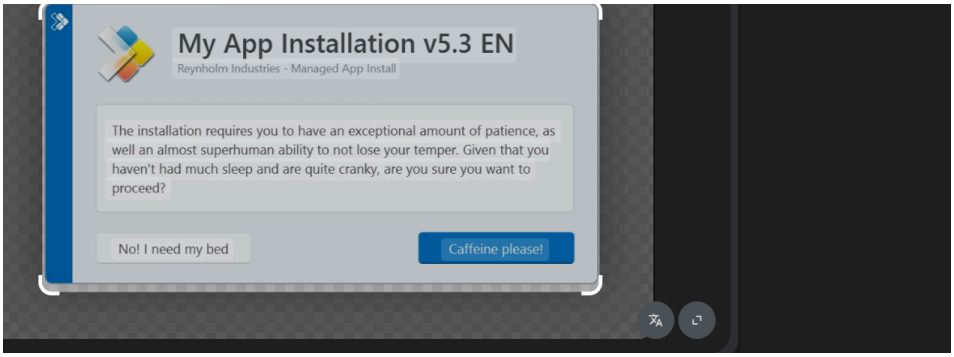
Output:



4.Installation Prompt:



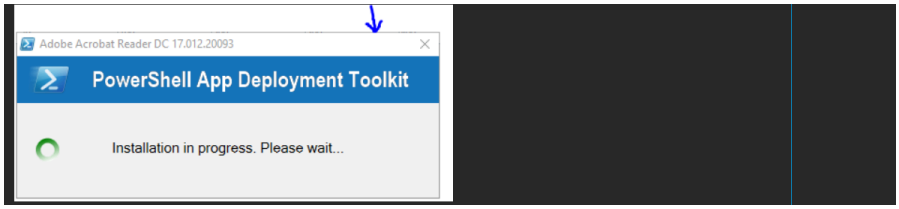
Output:



5.Installation Progress:

```
#
# MARK: Wrapper around Show-ADTInstallationProgress
#
#-----------------------------------------------------------------------------
        Show-ADTInstallationProgress
        #Displays a progress dialog with the status message 'Installation in Progress...'.
function Show-InstallationProgress
{
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$StatusMessage,

        [Parameter(Mandatory = $false)]
        [ValidateSet('Default', 'TopLeft', 'Top', 'TopRight', 'TopCenter', 'BottomLeft', 'Bottom', 'BottomRight')]
        [System.String]$WindowLocation,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$TopMost = $true,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Quiet,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$NoRelocation
    )

    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3

    # Announce overall deprecation before executing.
    Write-ADTLogEntry -Message "The function [$($MyInvocation.MyCommand.Name)] has been replaced by [Show-ADTInstallationProgress]. Please migrate your scripts to use the new fur
    if ($PSBoundParameters.ContainsKey('TopMost'))
    {
        $PSBoundParameters.Add('NotTopMost', !$PSBoundParameters.TopMost)
        $null = $PSBoundParameters.Remove('TopMost')
    }
    if ($PSBoundParameters.ContainsKey('Quiet'))
    {
        $PSBoundParameters.Add('InformationAction', [System.Management.Automation.ActionPreference]::SilentlyContinue)
        $null = $PSBoundParameters.Remove('Quiet')
```

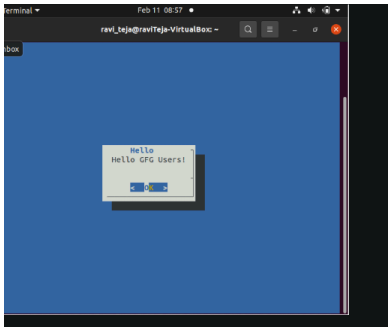
Output:



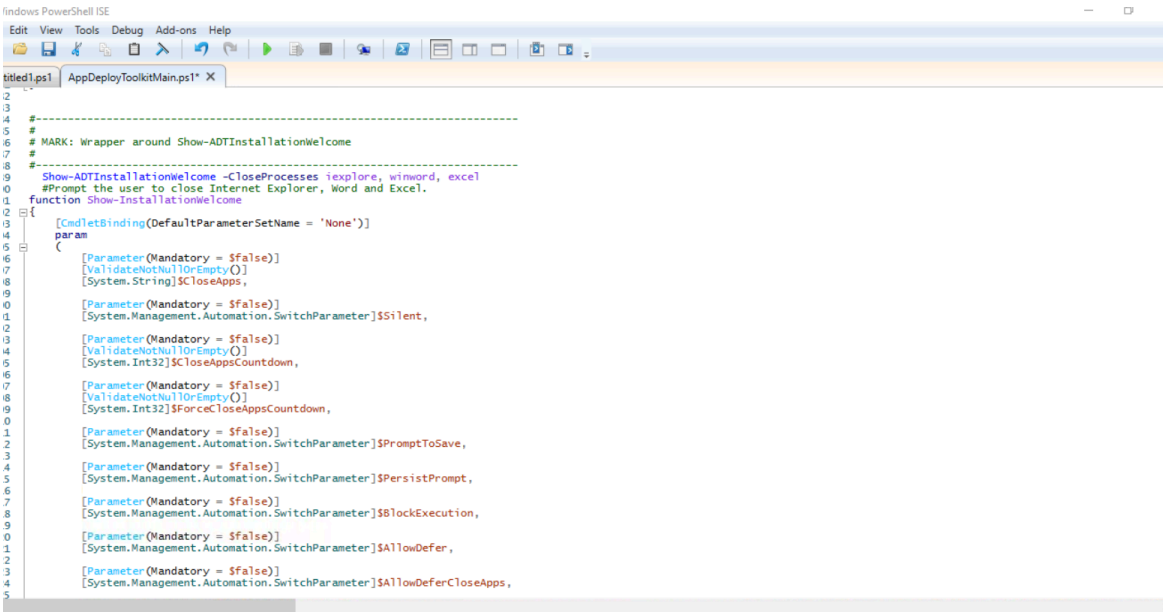
6.Show DialogBox:

```
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Show-ADTDialogBox
#
#-----------------------------------------------------------------------------
            Show-ADTDialogBox -Title 'Installation Notice' -Text 'Installation will take approximately 30 minutes. Do you wish to proceed?' -Buttons 'OKCancel' -DefaultButto
            #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 = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$TopMost
```

Output:

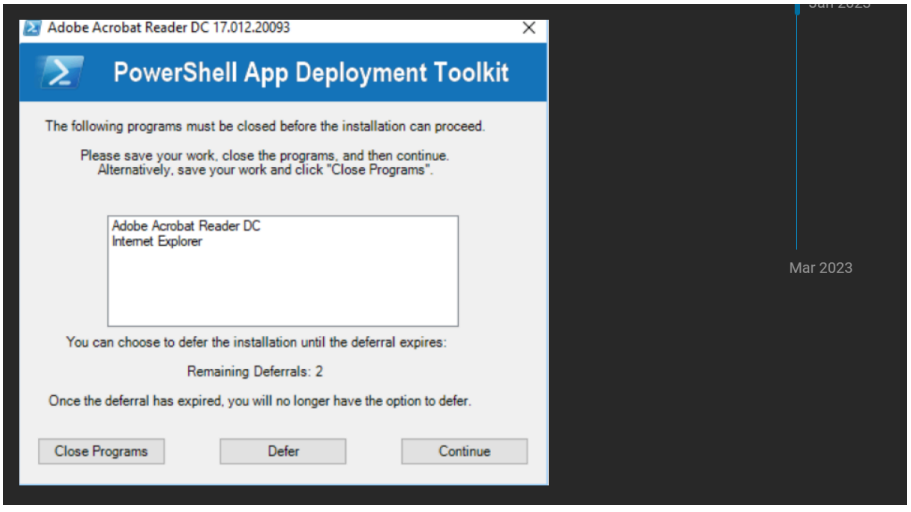


7.Show Installation Welcome:



```
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Show-ADTInstallationWelcome
#
#-----------------------------------------------------------------------------
  Show-ADTInstallationWelcome -CloseProcesses iexplore, winword, excel
  #Prompt the user to close Internet Explorer, Word and Excel.
function Show-InstallationWelcome
{
    [CmdletBinding(DefaultParameterSetName = 'None')]
    param
    (
        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$CloseApps,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$Silent,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Int32]$CloseAppsCountdown,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Int32]$ForceCloseAppsCountdown,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$PromptToSave,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$PersistPrompt,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$BlockExecution,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$AllowDefer,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$AllowDeferCloseApps,
```

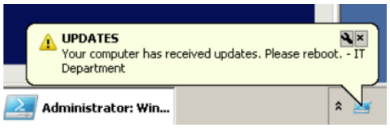
Output:



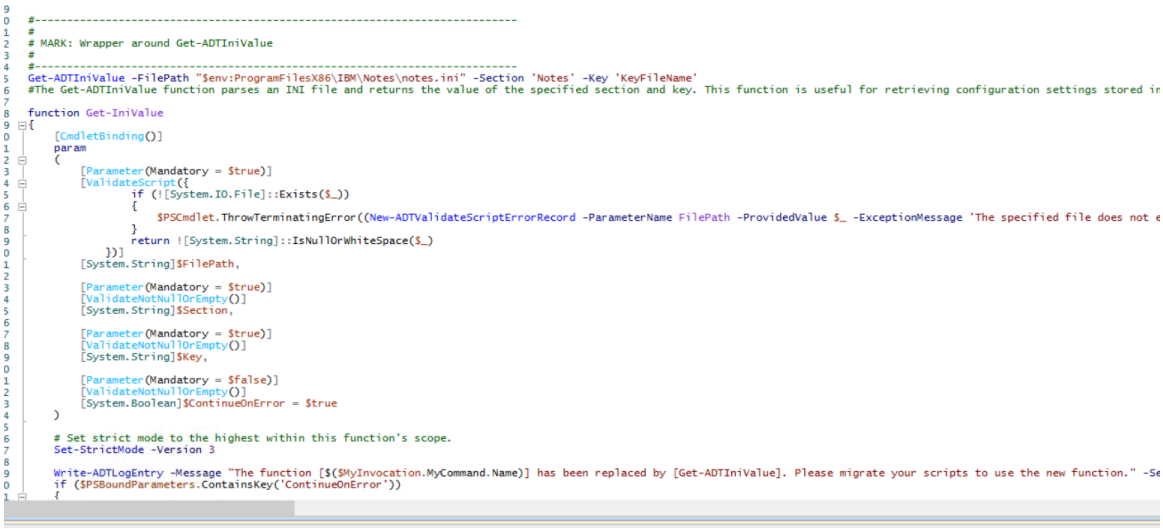
8.Show Balloon Tip:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Show-ADTBalloonTip
#
#-----------------------------------------------------------------------------
Show-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 Inivalue:



```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Get-ADTIniValue
#
#-----------------------------------------------------------------------------
Get-ADTIniValue -FilePath "$env:ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -Key 'KeyFileName'
#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

function Get-IniValue
{
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateScript({
                if (![System.IO.File]::Exists($_))
                {
                    $PSCmdlet.ThrowTerminatingError((New-ADTValidateScriptErrorRecord -ParameterName FilePath -ProvidedValue $_ -ExceptionMessage 'The specified file does not e
                }
                return ![System.String]::IsNullOrWhiteSpace($_)
            })]
        [System.String]$FilePath,

        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Section,

        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Key,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true
    )

    # 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 [Get-ADTIniValue]. Please migrate your scripts to use the new function." -Se
    if ($PSBoundParameters.ContainsKey('ContinueOnError'))
    {
```

10.Set IniValue:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Set-ADTIniValue
#
#-----------------------------------------------------------------------------
 Set-ADTIniValue -FilePath "$env:ProgramFilesX86\IBM\Notes\notes.ini" -Section 'Notes' -Key 'KeyFileName' -Value 'MyFile.ID'
 #Sets the 'KeyFileName' key in the 'Notes' section of the 'notes.ini' file to 'MyFile.ID'.
function Set-IniValue
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot sup
    [CmdletBinding(SupportsShouldProcess = $false)]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateScript({
                if (![System.IO.File]::Exists($_))
                {
                    $PSCmdlet.ThrowTerminatingError((New-ADTValidateScriptErrorRecord -ParameterName FilePath -ProvidedValue $_ -ExceptionMessage 'The specified file does not e
                }
                return ![System.String]::IsNullOrWhiteSpace($_)
            })]
        [System.String]$FilePath,

        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Section,

        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [System.String]$Key,

        [Parameter(Mandatory = $true)]
        [AllowNull()]
        [System.Object]$Value,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true
    )

    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
```

11.Start services and Dependencies:

```
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Start-ADTServiceAndDependencies
#
#-----------------------------------------------------------------------------
Start-ADTServiceAndDependencies -Name 'wuauserv'
#Starts the Windows Update service and its dependencies.
function Start-ServiceAndDependencies
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot suppo
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for back
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [Alias('Name')]
        [System.String]$Service,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$ComputerName,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$SkipServiceExistsTest,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$SkipDependentServices,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.TimeSpan]$PendingStatusWait,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$PassThru,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true
    )

    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
```

12.Stop Services and Dependencies:

```
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Stop-ADTServiceAndDependencies
#
#-----------------------------------------------------------------------------
  Stop-ADTServiceAndDependencies -Name 'wuauserv'
  #Stops the Windows Update service and its dependencies.
function Stop-ServiceAndDependencies
{
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseShouldProcessForStateChangingFunctions', '', Justification = "This compatibility wrapper function cannot suppo
    [System.Diagnostics.CodeAnalysis.SuppressMessageAttribute('PSUseSingularNouns', '', Justification = "This compatibility wrapper function cannot have its name changed for back
    [CmdletBinding()]
    param
    (
        [Parameter(Mandatory = $true)]
        [ValidateNotNullOrEmpty()]
        [Alias('Name')]
        [System.String]$Service,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$ComputerName,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$SkipServiceExistsTest,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$SkipDependentServices,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.TimeSpan]$PendingStatusWait,

        [Parameter(Mandatory = $false)]
        [System.Management.Automation.SwitchParameter]$PassThru,

        [Parameter(Mandatory = $false)]
        [ValidateNotNullOrEmpty()]
        [System.Boolean]$ContinueOnError = $true
    )

    # Set strict mode to the highest within this function's scope.
    Set-StrictMode -Version 3
```

13.Set Registry Key:

```powershell
#-----------------------------------------------------------------------------
#
# 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 su
    [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 = $false)]
        [ValidateSet('Binary', 'DWord', 'ExpandString', 'MultiString', 'None', 'QWord', 'String', 'Unknown')]
        [Microsoft.Win32.RegistryValueKind]$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
```

Ln 1183 Col 1

14.Remove Registry Key:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Remove-ADTRegistryKey
#
#-----------------------------------------------------------------------------
 Remove-ADTRegistryKey -Key 'HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce'
 #Deletes 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:

```powershell
#-----------------------------------------------------------------------------
#
# 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
```

Ln 1183 Col 1

16.Install MSUpdates:

```powershell
[}

#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Install-ADTMSUpdates
#
#-----------------------------------------------------------------------------
Install-ADTMSUpdates -Directory "$($adtSession.DirFiles)\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
    [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 functio
    try
    {
        Install-ADTMSUpdates @PSBoundParameters
    }
    catch
    {
        $PSCmdlet.ThrowTerminatingError($_)
    }
}

#-----------------------------------------------------------------------------
#
```

17.Set Active Setup:

```powershell
#-----------------------------------------------------------------------------
#
# 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 su
    [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 = $false)]
        [ValidateNotNullOrEmpty()]
        [System.String]$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:

```powershell
#-----------------------------------------------------------------------------
#
# MARK: Wrapper around Close-ADTInstallationProgress
#
#-----------------------------------------------------------------------------
Show-ADTInstallationProgress
#Uses the default status message from the strings.psd1 file.
function Close-InstallationProgress
{
    [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
    {
        $PSCmdlet.ThrowTerminatingError($_)
    }
}
```