```
<#
.SYNOPSIS
Reusable Windows update uninstaller for SCCM

.DESCRIPTION
Script is meant to be deployed once to DPs as a package, and then create programs for each update that needs to be uninstalled
Script accepts parameters and uninstalls Windows update via Dism.exe.

.PARAMETER PackageName
Package Identifier of Package found by running dism /online /get-packages

.PARAMETER Restart
If true, will allow for reboot of device. Will prompt user if loaded profile detected.

.PARAMETER UserMessage
If restart is selected, populates the title of the message box
EX: "Please restart your device to resolve the issue with INC0000001"
Default is "Please Restart to Finish Uninstalling Update"

.EXAMPLE
Run locally
powershell.exe -ExecutionPolicy Bypass -Command c:\Uninstall-WindowsPatch.ps1
-PackageName Package_for_KB982018~31bf3856ad364e35~amd64~~6.1.3.2

.EXAMPLE
Run from program in SCCM
"%WinDir%\SysNative\WindowsPowerShell\v1.0\powershell.exe -ExecutionPolicy Bypass
-Command .\Uninstall-WindowsPatch.ps1 -PackageName
Package_for_KB4073120~31bf3856ad364e35~amd64~~10.0.1.2072" -restart -UserMessage
'Per INC00002'


.NOTES
Version: 1.1
Author: Michael Fiorini
Updated: 8/16/19


#>


#Define parameters to be input through command line
```

```
param (
    [Parameter(Mandatory = $true)]
    [String]$PackageName,

    [switch]$Restart,

    [string]$UserMessage = "Please Restart to Finish Uninstalling Update"
    )


#Set Logging
$DeviceName = $env:COMPUTERNAME
$LogPath = $env:USERPROFILE #Enter path to log file directory
$AppName = 'Uninstall-WindowsPatch'
$LogName =  "$AppName`_$PackageName.log"
$Logfile = "$LogPath\$LogName"

#Create Log File if does not exist
If (-not(Test-Path -Path $Logfile)){
    New-Item -Path $LogPath -ItemType File -Name "$LogName"
    }

# Log Write function
function Write-Log {
    [CmdletBinding()]
    param(
        [Parameter()]
        [ValidateNotNullOrEmpty()]
        [string]$Message,

        [Parameter()]
        [ValidateNotNullOrEmpty()]
        [ValidateSet('Information','Warning','Error')]
        [string]$Severity = 'Information'
    )

    $Logtime = (Get-Date).ToString("yyyyMMdd_HHmmssfff")
    $LogEntry = "$LogTime`: $Severity`: $Message"
    #Encode to utf8 to allow combination with dism log
    $LogEntry | Out-File -FilePath $Logfile -Append -Encoding utf8
}#end Write-log function
```

```
#Test if package is installed
Write-Log -Message "Detecting if $PackageName is installed on $DeviceName" -Severity
Information

$Packages = dism /online /get-packages
$Package = $Packages | Select-string -Pattern "$PackageName"

If($Package -ne $null){
   Write-Log -Message "$PackageName was found on $DeviceName" -Severity Information
}#endif
else{
   Write-Log -Message "$PackageName was not found on $DeviceName" -Severity Warning
   Write-Log -Message "Exiting Script" -Severity Information
   [Environment]::Exit(0)
}#endelse


#Begin Uninstall process
Write-Log -Message "Initiating Uninstall of $PackageName on $DeviceName" -Severity
Information

#Uninstall Patch
Write-Log -Message "Running dism.exe /Online /Remove-Package
/PackageName:$PackageName /Quiet /norestart /logpath:$Logfile" -Severity Information

$DISMMessage = dism.exe /Online /Remove-Package /PackageName:$PackageName /Quiet
/norestart /logpath:$Logfile

#If Error, fail script and exit
$DISMError = $DISMMessage | Select-String -Pattern "Error"
If ($DISMError -like "Error*"){
   Write-Log -Message "Removing $PackageName failed on $DeviceName with error" -Severity
Error
   #Encode in utf 8 to allow for combination in dism log
   $DISMMessage | Out-File -FilePath $Logfile -Encoding utf8 -Append
   [Environment]::Exit(1)
}#endif
else {
   Write-Log -Message "dism.exe successfully removed $PackageName" -Severity Information
}#endelse


#Allow Reboot if restart flag is selected
```

```
If ($Restart){
   #Check if there is a user logged on
   Write-Log -Message "Checking for logged on users" -Severity Information
   $LoggedonUsers = Get-CimInstance -Class Win32_UserProfile -Filter "Loaded = True and
Special = False"

   #If users detected, prompt to reboot
   If ($LoggedonUsers -ne $null) {
      Write-Log -Message "User logins detected. Prompting to reboot." -Severity Information

      #Load VB
      [void][Reflection.Assembly]::LoadWithPartialName('Microsoft.VisualBasic')

      #Set message box properties
      $Message = "Windows requires a reboot to finish uninstalling an update. Please save your
work and select `"Yes`" to reboot now."
      $Title = $UserMessage

      #Prompt User to reboot
      $r = [Microsoft.VisualBasic.Interaction]::Msgbox($Message,
"YesNo,SystemModal,Exclamation", $Title)

      #If yes selected, reboot device
      If ($r -eq 'Yes'){
         Write-Log -Message "Rebooting $DeviceName to finish uninstall process" -Severity
Information
         #Shutdown with delay to allow script to exit successfully
         Shutdown /r /t 10 /f
      }#endif
      else{
         Write-Log -Message "User selected no. $DeviceName will not be rebooted" -Severity
Information
      }#endelse

   }#endif

   #If no users detected, reboot
   else {
      Write-log -Message "No users currently logged in detected" -Severity Information
      Write-Log -Message "Rebooting $DeviceName to finish uninstall process" -Severity
Information
      #Shutdown with delay to allow script to exit successfully
      Shutdown /r /t 10 /f
```

```
    }#endelse

}#endif
```