Assignment Date : 12-08-2025

Name : Lalit Joshi ,Email : [Joshilalit2275@gmail.com](mailto:Joshilalit2275@gmail.com)

UserID : **34734**

Topic 1: execute process in powershell

1. **Direct Execution:**

* Navigate to the folder (cd), then run the executable using .\YourFile.exe or by specifying the full path.

1. **Using Start-Process Cmdlet:**

* Runs processes asynchronously by default.
* Syntax: Start-Process -FilePath "path\to\file.exe".
* Options include -WindowStyle (e.g., Hidden), -ArgumentList (pass arguments), and -Credential (run as different user).
* Use -Wait if you want PowerShell to pause until the process ends.

1. **Running PowerShell Scripts:**

* Run scripts with .\YourScript.ps1.
* Run PowerShell as administrator to execute scripts with admin rights.

1. **Running Processes in Background:**

* Use Start-Process without -Wait or append & to run commands in the background.

1. **Using Invoke-Expression:**

* Executes expressions or scripts dynamically within the session.

1. **Running as Specific User:**

* Use functions like Execute-ProcessAsUser (from PSAppDeployToolkit) to run processes under specific user accounts.

1. **Execution Policy:**

* Adjust with Set-ExecutionPolicy to allow script execution.

**Important Notes:**

* Ensure you have proper permissions.
* Be cautious with untrusted scripts or executables.
* Use -Wait if the script needs to wait for process completion.

Topic 2 : Install-MSUpdates

**Purpose:**  
The Install-MSUpdates function installs all Microsoft Updates located in a specified directory. It supports update files with the extensions .exe, .msu, or .msp and searches the directory recursively.

**Syntax:**

powershell

CopyEdit

Install-MSUpdates -Directory <String> [CommonParameters]

**Key Points:**

* **Directory Parameter:**
  + Specifies the folder path containing update files.
  + Must be a valid directory path (not a single file).
  + Required parameter.
* Searches the given directory and all subdirectories for supported update files.
* Installs each found update file silently (depending on its installer’s capabilities).
* Does not return output objects; installation is performed without producing objects in the pipeline.
* Supports PowerShell common parameters like -Verbose for detailed output.

**Example Usage:**

powershell

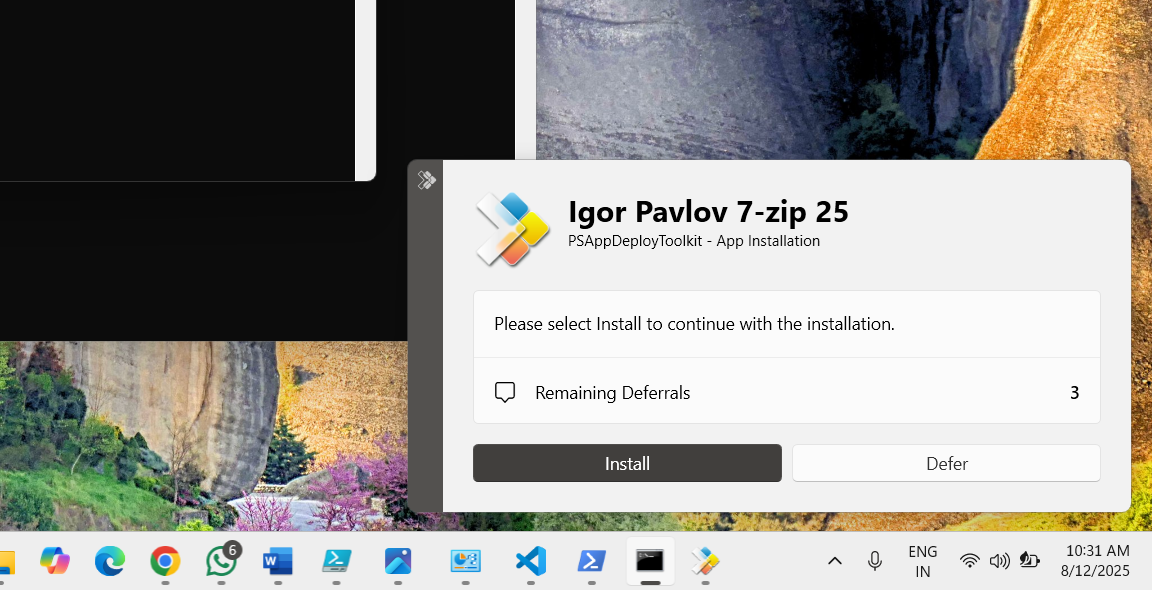
CopyEdit

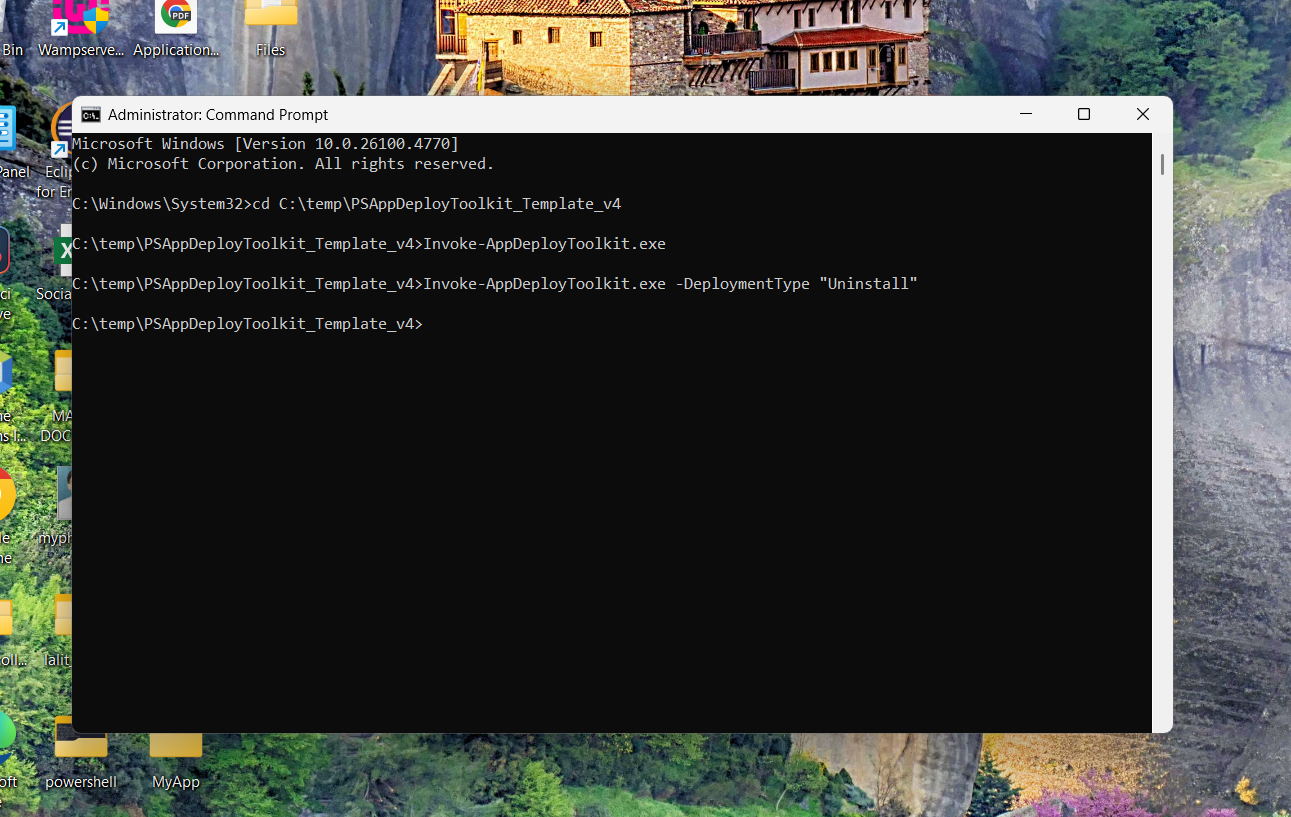
Install-MSUpdates -Directory "$dirFiles\MSUpdates" -Verbose

This command installs all Microsoft updates located in the MSUpdates folder under $dirFiles, displaying detailed progress messages.

**Important Notes:**

* Ensure you have administrative privileges before running.
* Only works if the Install-MSUpdates function is available (e.g., from PSAppDeployToolkit or a custom module).
* Does not accept pipeline input.





Topic 3: Active setup psadt

Active Setup in PSADT lets you run **per-user configurations or changes** automatically when a user logs in, even if the app was installed by a system account and doesn’t have built-in triggers like advertised shortcuts.

**How it works in PSADT:**

* **Registry Creation:** Set-ActiveSetup creates entries in HKLM:\SOFTWARE\Microsoft\Active Setup\Installed Components\<ProductCode> with the executable path (StubExePath), arguments, and a version number.
* **Login Trigger:** On login, Windows compares HKLM entries to HKCU. If HKCU is missing or outdated, the StubExePath runs.
* **Version Control:** Uses a timestamp (e.g., YYYYMMDDHHMMSS) to ensure re-execution when needed.
* **Execution Options:** Can run immediately for the current user or on next login.
* **Cleanup:** -PurgeActiveSetupKey removes entries from all HKCU hives during uninstall.

**Key Benefits:**

* **Standardized process** for per-user setup.
* **Automated** registry handling and version management.
* **Flexible customization** for path, arguments, and execution timing.
* **Clean removal** during uninstallation.

Topic 4: PSAppDeployToolkit (PSADT) – File & Folder Operations

**Purpose:**  
PSADT uses PowerShell cmdlets and its own custom functions to handle file/folder operations during software deployments, with built-in logging and error handling.

**1. Copying Files/Folders**

* **Copy-ADTFile** → Copies files with PSADT’s logging & error handling.

powershell

CopyEdit

Copy-ADTFile -Path "$dirSupportFiles\YourFile.txt" -Destination "$envProgramFiles\YourApp"

* **Copy-ADTFileToUserProfiles** → Copies files to all user profiles (supports recursion, Robocopy).

powershell

CopyEdit

Copy-ADTFileToUserProfiles -Path "$dirSupportFiles\UserSpecificFile.ini" -Destination "AppData\Roaming\YourApp"

* **Copy-Item** → Native PowerShell cmdlet for simple copy operations.

powershell

CopyEdit

Copy-Item -Path "C:\Source\Folder" -Destination "C:\Destination\Folder" -Recurse

**2. Deleting Files/Folders**

* **Remove-File** → Deletes specific files.

powershell

CopyEdit

Remove-File -Path "$envUserDesktop\OldShortcut.lnk"

* **Remove-Folder** → Deletes folders.

powershell

CopyEdit

Remove-Folder -Path "$envProgramFiles\OldApp"

* **Remove-Item** → Native PowerShell cmdlet for deleting files/folders (use -Force for hidden/read-only).

powershell

CopyEdit

Remove-Item -Path "C:\Temp\OldLog.log" -Force

**3. Moving Files/Folders**

* **Move-Item** → Native cmdlet for moving files/folders (no special PSADT function).

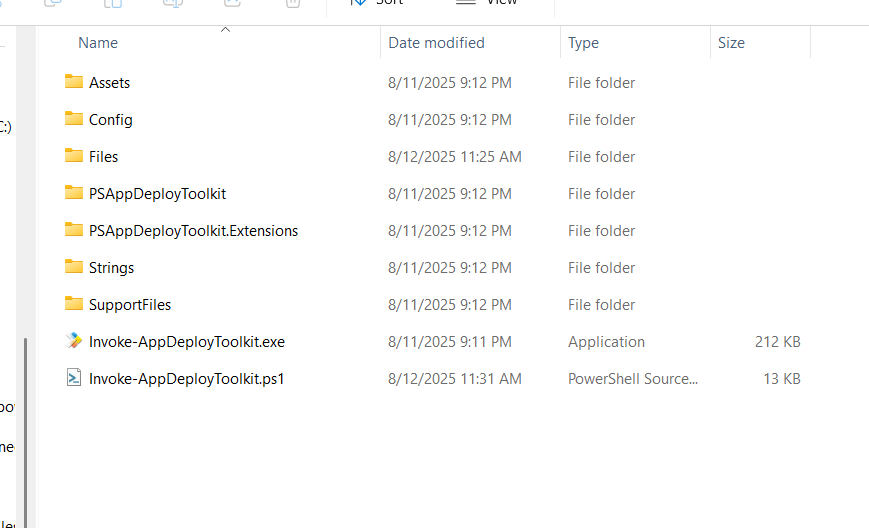
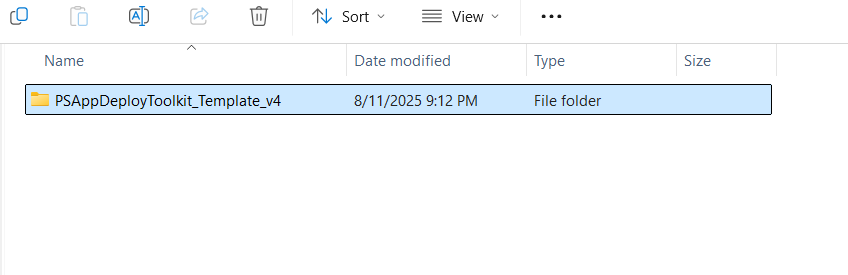
powershell

CopyEdit

Move-Item -Path "C:\Source\File.doc" -Destination "C:\Destination\Documents"

**Key Points**

* **Variables:** $dirSupportFiles, $envProgramFiles, $envUserDesktop, $envUserProfile help manage paths easily.
* **Logging:** PSADT functions log every action in deployment logs.
* **Error Handling:** Built-in handling, with options like -ContinueOnError for custom behavior.



Topic 5:Important Commands of Psadt

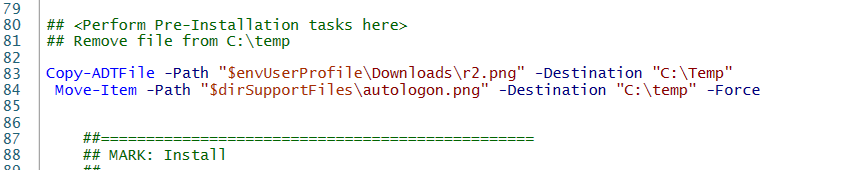
**1. Copying Files and Folders:**

**Copy-ADTFile**  
Copies files during PSADT deployment with error handling and logging.

powershell

CopyEdit

Copy-ADTFile -Path "$dirSupportFiles\YourFile.txt" -Destination "$envProgramFiles\YourApp"

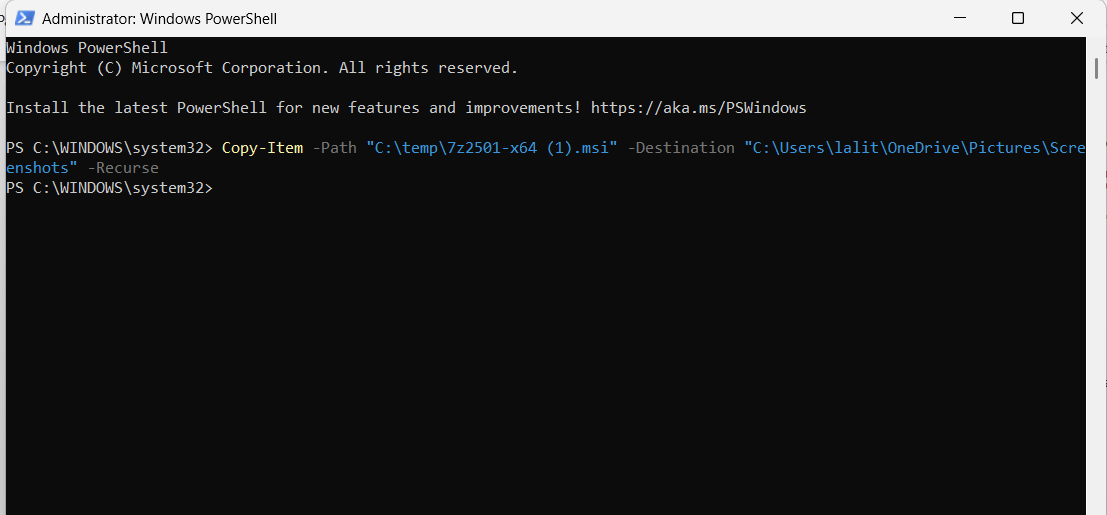


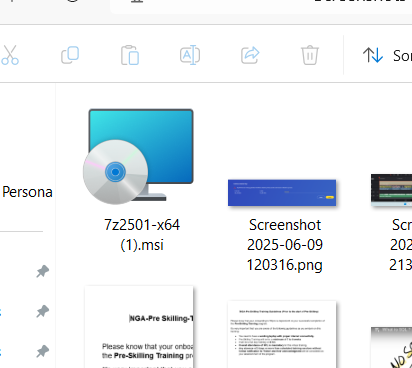
**2.Copy-Item**  
Native PowerShell cmdlet for basic copy operations.

powershell

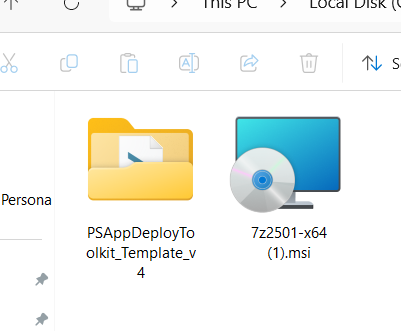
CopyEdit

Copy-Item -Path "C:\Source\Folder" -Destination "C:\Destination\Folder" -Recurse





**Destionation Loaction Folder**

****

**Source Folder**

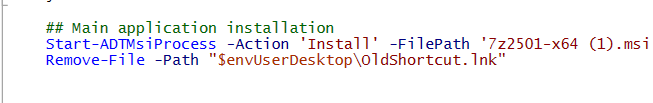
**3. . Deleting Files and Folders:**

**Remove-File  
PSADT function to delete specific files.**

**powershell**

**CopyEdit**

**Remove-File -Path "$envUserDesktop\OldShortcut.lnk"**

****

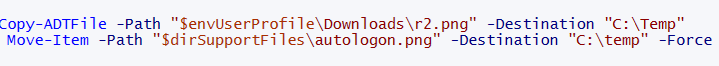
4. **Moving Files and Folders:**

**Move-Item**  
Native PowerShell cmdlet to move files or folders.

powershell

CopyEdit

Move-Item -Path "C:\Source\File.doc" -Destination "C:\Destination\Documents"



5. **Execute-ADTMSI -Action**

The Execute-ADTMSI function in PSAppDeployToolkit is used to run MSI (Microsoft Installer) packages during software deployment. The -Action parameter specifies what operation to perform on the MSI package.

**Syntax:**

powershell

CopyEdit

Execute-ADTMSI -Action <Install|Uninstall|Repair> -Path "<PathToMSI>" [-Parameters "<MSI Parameters>"] [-LogName "<LogFileName>"] [-ContinueOnError]

**Parameters:**

* -Action  
  Defines the MSI operation.
  + Install — Installs the MSI package.
  + Uninstall — Uninstalls the MSI package.
  + Repair — Repairs the existing MSI installation.
* -Path  
  Full path to the MSI installer file.
* -Parameters (Optional)  
  Additional command-line parameters to pass to msiexec.exe (e.g., silent install switches like /qn).
* -LogName (Optional)  
  Specifies the name of the installation log file.
* -ContinueOnError (Optional)

