**All GET Commands:**

* **Get-NetIPConfiguration:**

*It* ***shows the network IP configuration*** *of your computer. This includes details like your* ***IP address, subnet mask, default gateway, and DNS servers.*** *It helps you see how your computer is connected to the network.*

* Get-NetIPConfiguration | Out-File D:\wipropowershell\datafiles.txt -Append
* Get-NetIPConfiguration | Out-File D:\aa\wiprofiledwc\netfileipconfig
* Get-NetIPConfiguration | Out-File D:\Demo\datefile.txt -Append
* **Get-PSDrive:**

*It* ***lists all the drives*** *available in your PowerShell session. This includes* ***hard drives, network drives, and special PowerShell drives*** *like the Registry or Environment variables. It shows the drive name, used space, free space, and the root path.*

* Get-PSDrive -PSProvider FileSystem | Out-File D:\wipropowershell\datefilea.txt -Append
* Get-PSDrive -PSProvider FileSystem | Out-File D:\Demo\datefile.txt -Append
* Get-PSDrive -PSProvider FileSystem | Out-File D:\Demo\datefile.txt -Append
* Get-PSDrive -PSProvider FileSystem | Out-File D:\wipropowershell\datafiles.txt -Append
* **Get-ChildItem:**

*It* ***lists the files and folders*** *inside a directory (folder). Think of it like the dir command in Windows or the ls command in Linux. By default, it shows everything in the current folder. You can use it to see what’s inside a folder, including subfolders if you want.*

* Get-ChildItem | Out-File D:\Demo\datefile.txt -Append
* **Get-Member:**

***Get-Member*** *helps you explore what properties and methods objects have.Using* ***Out-File -Append*** *saves the output to a file without deleting existing content.*

* Get-Member | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Member -ListAvailable Name | Out-File D:\Demo\opclfilecommands.txt -Append
* **Get-Date:**

*The* ***Get-Date*** *command in PowerShell is used to* ***get the current date and time****. In simple terms, it tells you what the date and time are right now on your computer.*

* Get-Date | Get-Member | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Date | Select-Object –Property Second
* Get-Date | Select-Object –Property TimeofDay
* **Get-Command:**

***Get-Command*** *is a PowerShell command that shows you* ***all the commands, cmdlets, functions, scripts, and executables available*** *in your PowerShell session. It helps you* ***find commands you can use*** *and get details about them.*

* Get-Command \*hotfix\* Out-File D:\Demo\opfilecommands.txt -Append
* Get-Command | Out-File D:\Demo\datefile.txt -Append
* **Get-Hotfix:**

***Get-Hotfix*** *is a PowerShell command that lists all* ***Windows updates and patches*** *(hotfixes) installed on your computer. It helps you* ***see which updates have been installed****, including security patches, bug fixes, and service packs*.

* Get-Hotfix | Get-Member
* Get-Hotfix | Select-Object –Property InstalledBy | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Hotfix | Select-Object –Property HotFixID,InstalledOn,InstalledBy | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Hotfix | Select-Object –Property HotFixID,@{n='HotFixAge';e={(New- TimeSpan –Start $PSItem.InstalledOn).Days}},InstalledBy

help \*rule\*

* **Get-NetNeighbor**

***Get-NetNeighbor*** *is a PowerShell command that shows the* ***neighbor (ARP) table*** *on your computer. This table maps* ***IP addresses to MAC addresses*** *on your local network.It helps you* ***see devices connected to your local network****, along with their IP and MAC addresses.*

* Get-NetNeighbor | Sort-Object –Property State | Select-Object –Property IPAddress,State | Format-Wide -GroupBy State - | Out-File D:\Demo\cd.txt -Append
* Get-NetFirewallRule:

***Get-NetFirewallRule*** *is a PowerShell command used to* ***view firewall rules*** *on a Windows computer.It helps you* ***see which firewall rules are active****, what programs or ports they apply to, and whether they allow or block traffic.*

* Get-NetFirewallRule | Out-File D:\Demo\Ipfirewallconfig.txt -Append
* Get-NetIPConfiguration | Out-File D:\Demo\netipconfig.txt -Append
* Help Get-NetFirewallRule | Out-File D:\Demo\netipconfig.txt -Append
* Help Get-NetFirewallRule –ShowWindow
* Get-NetFirewallRule –Enabled True | Out-File D:\Demo\a.txt -Append
* Get-NetFirewallRule –Enabled True | Format-Table -wrap | Out-File D:\Demo\b.txt -Append
* Get-NetFirewallRule | Format-Table -wrap | Out-File D:\Demo\abc.docx -Append
* Get-NetFirewallRule –Enabled True | Select-Object –Property DisplayName,Profile,Direction,Action | Sort-Object –Property Profile, DisplayName | Format-Table -wrap | Out-File D:\Demo\ad.txt -Append
* Get-NetFirewallRule –Enabled True

100 -gt 10

500 -le 100

'hello' -eq 'HELLO'

'hello' -ceq 'HELLO'

* **Get-Service Commands:**

***Get-Service*** *is a PowerShell command that shows the* ***status of Windows services*** *— these are background programs that help the system or apps run properly. These commands help you* ***see which services are running or stopped****, and you can save that info to a file if needed.*

* Get-Service | Where status -ne Running | Out-File D:\Demo\a.txt -Append
* Get-Service | Where status -ne Running
* Get-Service | ForEach Status
* Get-Service | ForEach Displayname
* Get-Service | Get-Member | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Service | Out-File D:\Demo\opfilecommands.txt -Append
* Get-Service| Out-File D:\aa\wiprofiledwc\processfile -Append
* **Get-EventLog Commands:**

***Get-EventLog*** *is a PowerShell command that lets you* ***read Windows event logs****, which record system errors, warnings, and startup messages. These commands are used to* ***view and filter event logs*** *to check system activity, errors, or events, like a computer logbook.*

* Get-EventLog –List
* Get-EventLog –List | Where Log –eq 'System'
* Get-EventLog –List | Where Log –eq 'System'
* **Get-Process Commands:**

*Get-Process is a PowerShell command that shows a list of running programs and background processes on your computer.*

*These commands help you see, filter, and stop running processes — similar to what Task Manager does, but in a script or terminal.*

* Get-Process | Where CPU -gt 100
* Get-Process -Name EXCEL | Stop-Process
* Stop-Process -Name
* Get-Process | ConvertTo-HTML
* Get-Process | ConvertTo-HTML | Out-File D:\Demo\htmldata.txt
* Get-Process | Sort-Object CPU -Descending | Select-Object -First 5 | Format-Table Name, CPU

**All Execute-MSI Commands:**

* **Execute-MSI commands:**

*Execute-MSI is a* ***PowerShell function used to install, uninstall, or patch Windows Installer packages*** *—* .msi *and* .msp *files. It’s commonly used in* **software deployment scripts** *(like with MDT, SCCM, or PSAppDeployToolkit). Execute-MSI lets you manage software installations and updates silently and automatically.*

* Execute-MSI -Action 'Install' -Path 'Adobe\_FlashPlayer\_11.2.202.233\_x64\_EN.msi'
* 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'
* Execute-MSI -Action 'Uninstall' -Path '{26923b43-4d38-484f-9b9e-de460746276c}'
* Execute-MSI -Action 'Patch' -Path 'Adobe\_Reader\_11.0.3\_EN.msp'
* Execute-MSI -Action 'Patch' -Path 'Adobe\_Reader\_11.0.3\_EN.msp'
* Execute-MSI -Action Install -Path $AppMSIName -SkipMSIAlreadyInstalledCheck -ContinueOnError $False -LogName "${AppMSIName}\_MSI"
* Execute-MSI -Action 'Install' -Path 'Adobe\_FlashPlayer\_11.2.202.233\_x64\_EN.msi'
* 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'
* Execute-MSI -Action 'Uninstall' -Path '{26923b43-4d38-484f-9b9e-de460746276c}'
* Execute-MSI -Action 'Patch' -Path 'Adobe\_Reader\_11.0.3\_EN.msp'
* Execute-MSI -Action 'Patch' -Path 'Adobe\_Reader\_11.0.3\_EN.msp'
* Execute-MSI -Action Install -Path $AppMSIName -SkipMSIAlreadyInstalledCheck -ContinueOnError $False -LogName "${AppMSIName}\_MSI"
* **Execute process Commands:**

*Execute-Process is a PowerShell command (often used in deployment scripts) that runs other programs or installers silently or with custom settings.*

*It’s used to* ***install, uninstall, or run software automatically****, often* ***without showing windows or asking the user anything****.*

* Execute-Process -Path 'uninstall\_flash\_player\_64bit.exe' -Parameters '/uninstall' -WindowStyle 'Hidden'
* Execute-Process -Path "$dirFiles\Bin\setup.exe" -Parameters '/S' -WindowStyle 'Hidden'
* Execute-Process -Path 'setup.exe' -Parameters '/S' -IgnoreExitCodes '1,2'
* Execute-Process -Path 'setup.exe' -Parameters "-s -f2`"$configToolkitLogDir\$installName.log`""
* Execute-Process -Path 'setup.exe' -Parameters "/s /v`"ALLUSERS=1 /qn /L\* `"$configToolkitLogDir\$installName.log`"`""

* **Execute MSP Commands:**

*They are used to install software updates or patches, often silently (without showing a user interface).In* ***Windows****, an* ***MSP file*** *is a* ***Microsoft Patch file*** *— it's used to apply updates to software installed via Windows Installer (like .msi files).*

* Execute-MSP -Path 'Adobe\_Reader\_11.0.3\_EN.msp'
* Execute-MSP -Path 'AcroRdr2017Upd1701130143\_MUI.msp' -AddParameters 'ALLUSERS=1'