Windows Scripting Cheatsheet

A quick reference guide for Windows scripting using PowerShell and Batch scripting, covering essential commands, syntax, and automation techniques.

PowerShell Basics

Core Commands (Cmdlets)

`Get-Command`	Lists all available PowerShell commands (cmdlets).
`Get-Help <command>`	Displays help information for a specific command.
`Set-ExecutionPolicy`	Configures the PowerShell execution policy (e.g., `RemoteSigned`).
`Get-Process`	Retrieves information about running processes.
`Stop-Process`	Stops a running process (e.g., `Stop-Process -Id <PID>`).
`Get-Service`	Retrieves information about Windows services.
`Start-Service`	Starts a Windows service.
`Stop-Service`	Stops a Windows service.
`Restart-Service`	Restarts a Windows service.

Variables and Operators

`$`	Indicates a variable (e.g., `$name = "John"`).
`=`	Assignment operator (e.g., `$x = 10`).
`-eq`, `-ne`, `-gt`, `-lt`	Equality, inequality, greater than, and less than operators.
`-and`, `-or`, `-not`	Logical AND, OR, and NOT operators.
`+`, `-`, `*`, `/`	Arithmetic operators (addition, subtraction, multiplication, division).
`+=`, `-=`	Shorthand for incrementing/decrementing the value of a variable.

Basic Syntax & Structures

- `if ($condition) { ... } elseif ($condition) { ... } else { ... }` - Conditional statement.
- `for ($i = 0; $i -lt 10; $i++) { ... }` - For loop.
- `foreach ($item in $collection) { ... }` - Foreach loop.
- `while ($condition) { ... }` - While loop.
- `function My-Function { ... }` - Function definition.

PowerShell Advanced

Working with Objects

`Get-Member`	Displays the properties and methods of an object.
`Select-Object`	Selects specific properties of an object (e.g., `Get-Process | Select-Object Name, Id`).
`Where-Object`	Filters objects based on a condition (e.g., `Get-Process | Where-Object {$_.CPU -gt 1}`).
`ForEach-Object`	Performs an action on each object in a collection (e.g., `Get-Process | ForEach-Object {Write-Host $_.Name}`).
`Export-Csv`	Exports objects to a CSV file (e.g., `Get-Process | Export-Csv -Path processes.csv -NoTypeInformation`).
`Import-Csv`	Imports objects from a CSV file (e.g., `$data = Import-Csv -Path data.csv`).

Modules and Script Execution

`Import-Module`	Imports a PowerShell module (e.g., `Import-Module ActiveDirectory`).
`Get-Module`	Lists imported modules.
`. <script_path>`	Executes a PowerShell script in the current scope.
`& <script_path>`	Executes a PowerShell script in a new scope.
`. <script_path>`	Executes a PowerShell script.
`Publish-Module`	Publishes a module to a repository.

Error Handling

- `try { ... } catch { ... } finally { ... }` - Try-Catch-Finally block for error handling.
- `$ErrorActionPreference = 'Stop'` - Sets the error action preference to stop execution on error.
- `Write-Error "Error message"` - Writes an error message.

Batch Scripting Basics

Basic Commands

`echo`	Displays text on the console.
`@echo off`	Turns command echoing off (usually at the beginning of the script).
`pause`	Pauses script execution until a key is pressed.
`cls`	Clears the console screen.
`title`	Sets the title of the console window.
`exit`	Exits the script.
`rem` or `::`	Comment a line (rem is compatible with older versions).

Variables and Operators

`set <variable>=<value>`	Sets a variable (e.g., `set name=John`).
`%<variable>%`	Accesses a variable's value (e.g., `echo %name%`).
`==`, `EQU`, `NEQ`, `GTR`, `LSS`	Equality, not equal, greater than, and less than operators (string comparison).
`if <condition> ( ... ) else ( ... )`	Conditional statement.
`for %%i in (<set>) do ( ... )`	For loop (e.g., `for %%i in (a b c) do echo %%i`). Use `%i` in command line
`goto <label>`	Jumps to a labeled section of the script.

File and Directory Operations

`dir`	Lists files and directories in the current directory.
`cd`	Changes the current directory (e.g., `cd C:\Windows`).
`mkdir` or `md`	Creates a new directory (e.g., `mkdir NewDirectory`).
`rmdir` or `rd`	Removes a directory (e.g., `rmdir NewDirectory`).
`copy`	Copies files (e.g., `copy file.txt C:\Backup`).
`del`	Deletes files (e.g., `del file.txt`).

Batch Scripting Advanced

Conditional Statements and Loops

`if <condition> ( ... ) else ( ... )` - Conditional execution.
Example:

```
if "%ERRORLEVEL%" == "0" (
  echo Success
) else (
  echo Failure
)```

`for %%i in (<set>) do ( ... )` - Looping through a set of items.
Example:

```
for %%i in (*.txt) do (
  echo Processing %%i
)```

`goto <label>` - Jump to a specific label in the script.
Example:

```
:start
echo Starting...
goto end
:end
echo Ending...
```

Working with Strings

`%variable:~start,length%`	Substring extraction from a variable (e.g., `%string:~0,5%`).
`set variable=%variable:old=new%`	String replacement (e.g., replace 'old' with 'new' in %variable%).
`set variable=%variable:<search>=%`	Delete search string

Calling External Programs

`start <program>` - Starts an external program.
Example:

```
start notepad.exe file.txt```

`call <batch_file>` - Calls another batch file.
Example:

```
call other_script.bat```