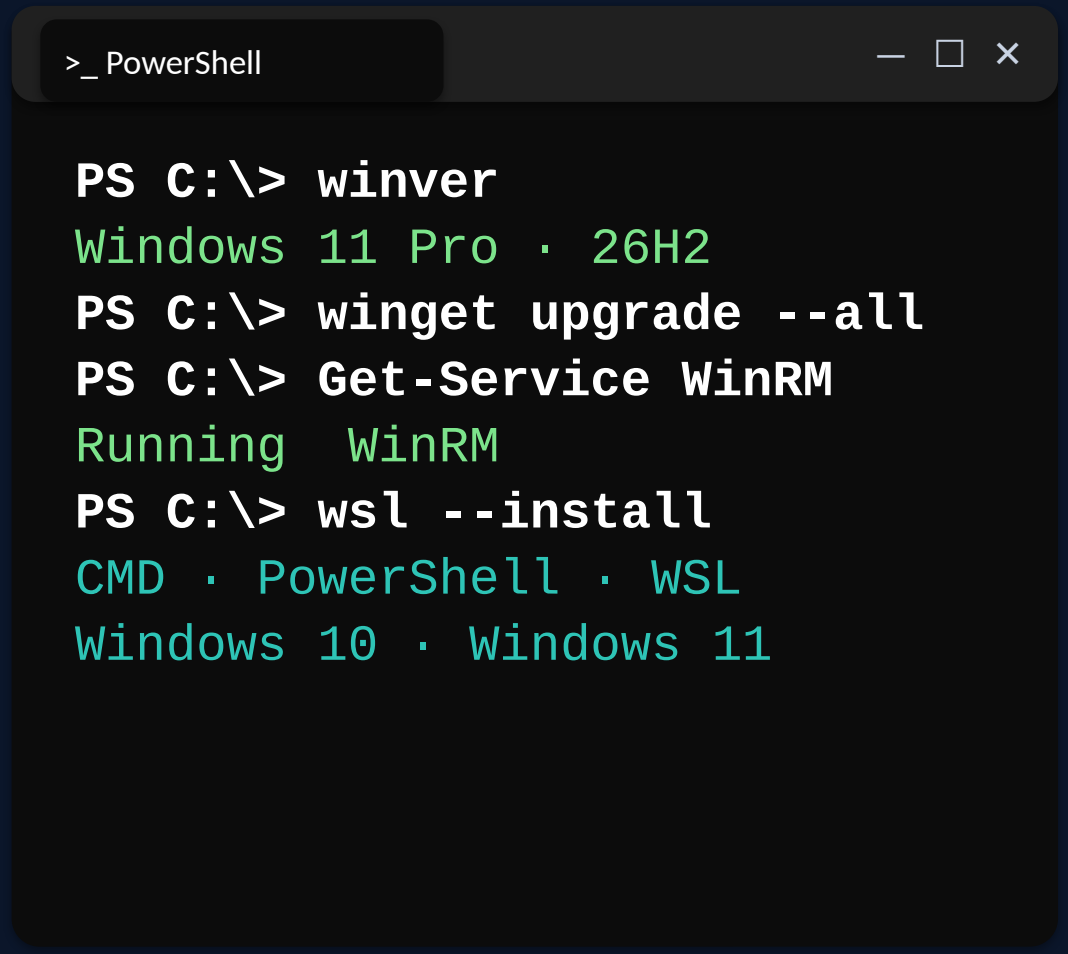


LESSON 4 OF 12

Windows 10 & 11

Desktop · Workstation · Laptop

Command Line: CMD, PowerShell, Windows Terminal & NTFS

Full Course · Windows 11 26H2 · Windows 10 22H2

What You Will Learn in Lesson 4

- The shells: CMD, Windows PowerShell 5.1, PowerShell 7 and WSL
- Windows Terminal: tabs, profiles, admin sessions and sudo
- CMD vs PowerShell vs Linux: the same tasks side by side
- PowerShell thinking: Verb-Noun, objects and the pipeline
- Paths, drives and the NTFS file system
- Working with files in CMD and PowerShell, including robocopy
- Your first batch file and PowerShell script

The Shells in Windows

Shell	Program	Notes
Command Prompt	cmd.exe	Classic; batch files (.bat, .cmd); still everywhere
Windows PowerShell 5.1	powershell.exe	Built in; frozen at 5.1; still used by many scripts
PowerShell 7.6 (LTS)	pwsh.exe	Modern, open source, on .NET 10; Windows, macOS, Linux
WSL (bash)	wsl.exe	A real Linux shell and distro (Lesson 6)
Windows Terminal	wt.exe	The window that hosts them all; default on Windows 11

Install PowerShell 7 and Check Versions

```
PS C:\> winget install --id Microsoft.PowerShell --source winget
PS C:\> pwsh                                    # start PowerShell 7
PS C:\> $PSVersionTable.PSVersion
Major  Minor  Patch
7      6      4

C:\> ver                                        # CMD: Windows version
Microsoft Windows [Version 10.0.26300.xxxx]
```

Version numbers change with updates — yours may be newer.

Same Task, Three Shells

Task	CMD	PowerShell	Linux bash
List files	dir	Get-ChildItem (ls, dir)	ls -l
Change folder	cd	Set-Location (cd)	cd
Copy / move	copy / move	Copy-Item / Move-Item	cp / mv
Delete	del, rmdir	Remove-Item (rm)	rm, rmdir
Show a file	type	Get-Content (cat)	cat
Search text	findstr	Select-String	grep
Clear screen	cls	Clear-Host (cls)	clear

Thinking in PowerShell

- Commands are Verb-Noun cmdlets: Get-Service, Stop-Process, New-Item
- Output is objects with properties, not just text
- The pipeline | passes objects to the next command
- Get-Command finds commands; Get-Help explains them (-Examples)
- Get-Member shows the properties and methods of any object
- -WhatIf previews a change; -Confirm asks first

The Pipeline in Action

```
PS C:\> Get-Process | Sort-Object CPU -Descending | Select-Object -First 5 Name, CPU, Id

PS C:\> Get-Service | Where-Object Status -eq 'Running' | Measure-Object

PS C:\> Get-ChildItem C:\Windows -File | Measure-Object Length -Sum

PS C:\> Get-Command *service*              # find commands
PS C:\> Get-Help Get-Service -Examples     # learn by example
PS C:\> Get-Service WinRM | Get-Member     # what is inside the object?
```

Paths, Drives and NTFS

Topic	What to know
Drive letters	C: is Windows; D:, E: for other disks, USB, DVD
Paths	Backslash: C:\Users\wahid; network: \\server\share (UNC)
Case	Not case-sensitive, but case-preserving
NTFS features	Permissions (ACLs), journaling, compression, EFS, hard links
Other file systems	ReFS (workstations, servers), FAT32 and exFAT (USB drives)
Gotchas	Reserved names (CON, PRN, NUL); 260-character path limit by default

Files in CMD (including robocopy)

```
C:\> mkdir C:\Lab\Reports & cd /d C:\Lab
C:\Lab> echo Hello Windows > notes.txt
C:\Lab> copy notes.txt Reports\           & dir /a Reports
C:\Lab> type notes.txt                     & attrib +h notes.txt
C:\Lab> tree /f C:\Lab

C:\> robocopy C:\Lab D:\Backup\Lab /E /R:1 /W:1 /LOG:C:\robocopy.log
#  /E copy subfolders   /MIR mirror (DELETES extra files at the destination!)
```

Files in PowerShell

```
PS C:\> New-Item -ItemType Directory C:\Lab\Scripts
PS C:\> Set-Content C:\Lab\todo.txt 'Learn PowerShell'
PS C:\> Add-Content C:\Lab\todo.txt 'Practice every day'
PS C:\> Get-Content C:\Lab\todo.txt
PS C:\> Copy-Item C:\Lab C:\Lab2 -Recurse
PS C:\> Remove-Item C:\Lab2 -Recurse -WhatIf      # preview first!
PS C:\> Test-Path C:\Lab\todo.txt
PS C:\> Get-Content C:\Windows\Logs\CBS\CBS.log -Tail 20 -Wait   # follow a log
```

Your First Scripts

```
:: hello.bat  — a batch file
@echo off
echo Hello %USERNAME%, today is %DATE%

# hello.ps1  — a PowerShell script
param([string]$Name = $env:USERNAME)
"Hello $Name, you have $((Get-Process).Count) processes running."

PS C:\Lab> Get-ExecutionPolicy                    # RemoteSigned on Windows 11
PS C:\Lab> .\hello.ps1 -Name Student              # dot-backslash runs it
```

Windows Terminal Like a Pro

- Profiles: PowerShell 7, Windows PowerShell, CMD, WSL, Azure Cloud Shell
- Settings > Startup: choose the default profile (PowerShell 7)
- Ctrl + Shift + T new tab; Alt + Shift + D split pane
- Right-click a profile > Run as administrator
- sudo for Windows: Settings > System > Advanced > Enable sudo
- Your profile script: notepad $PROFILE (aliases, prompt, modules)

Summary & Homework

- CMD for classic tasks; PowerShell 7 for modern administration
- PowerShell = Verb-Noun + objects + pipeline; use Get-Help and -WhatIf
- Know NTFS, paths and UNC; robocopy for reliable copies
- Scripts: .bat for CMD, .ps1 for PowerShell (execution policy)
- Homework: install PowerShell 7 and make it the Terminal default
- Homework: write hello.ps1 and list your top 5 CPU processes

Thank You! 🙏

Free videos & PDFs: www.MyWebUniversity.com

Slides for this lesson: MyWebUniversity.com/presentations

YouTube: @MyWebUniversity · @MyWebUniversity-Dari

Wahidullah Lutfy — Founder and Director of IT of MyWebUniversity, and CS teacher

Book: The Direct Path to Linux Ubuntu (Amazon · Barnes & Noble)

Learn it, then pass it on.