Windows 11 Fundamentals

A Complete 12-Chapter Operating Systems Course

Topics covered:

The interface, installation & TPM/Secure Boot · File Explorer & NTFS

Accounts, Windows Hello & UAC · the Settings app · installing & managing software

Networking & profiles · updates & Storage Sense · built-in security · PowerShell

Accessibility & productivity features

Capstone: a full personal workstation setup, chapter by chapter

Exercises: 36 hands-on scenarios with worked solutions

Format: A4 · Dark-theme code examples

Philip Osztromok · Generated with Claude

Table of Contents

CHAPTER 1 OF 12

What Windows 11 Is & The Interface Tour

Windows 11 Fundamentals

Chapter 1 · What Windows 11 Is & The Interface Tour

Windows remains the desktop operating system most people, in most jobs, actually sit in front of every day — including, most likely, yours. *Installing and Configuring Linux* already covers the site's other major general-purpose desktop OS in real depth; a dedicated macOS course is reserved on the bucket list but not yet started. This course exists to give Windows 11 the same treatment: not a tour of what's new for its own sake, but real, day-to-day competence — the kind that turns into genuine day-job skill once Windows 11 Troubleshooting & Administration builds on it.

The Interface Tour

Windows 11's interface is a real, visible redesign from Windows 10 — not just a fresh coat of paint. Four changes are worth knowing by name before anything else:

- **Start Menu** — centered on the taskbar by default (Windows 10's was left-aligned), and rebuilt around pinned apps and a separate "Recommended" section rather than the old Live Tiles grid.
- **Taskbar** — icons centered by default (movable back to the left in Settings), a redesigned System Tray, and — a genuine limitation worth knowing rather than discovering by accident — no built-in support for a vertical taskbar or dragging files directly onto a taskbar icon, both of which worked in Windows 10.
- **Widgets** — a slide-out panel (news, weather, calendar) accessed from the taskbar, personalized and, notably, powered by a Microsoft account sign-in even on an otherwise local-account machine.
- **Snap Layouts** — hover over any window's maximize button (or press `Win + Z`) to reveal a set of predefined window-arrangement grids, then click a slot to snap the window into it. Genuinely useful multitasking, not a gimmick — worth building into real muscle memory.

Settings vs. Control Panel — This Course's Own Throughline

Windows 11 has spent years migrating settings out of the legacy **Control Panel** and into the modern **Settings** app — but that migration still isn't complete. Some things (advanced network adapter properties, certain legacy device configuration, a handful of user-account edge cases) genuinely only exist in Control Panel today. This isn't an oversight to work around once and forget; it's the first concrete instance of something that runs through this entire course and its own sequel:

THE PATTERN THIS WHOLE TRACK IS BUILT AROUND

Windows exposes the same underlying system state through **at least three different interfaces** — a modern GUI (Settings), a legacy GUI (Control Panel), and a scriptable command line (PowerShell, introduced in Chapter 10) — and Windows 11 Troubleshooting & Administration adds a fourth, deeper layer still: the Registry and Group Policy. Knowing which interface to reach for, and why the older ones haven't simply been deleted, is real support-engineer judgment, not trivia.

Windows 11 Among the Site's Other Operating Systems

	INTERFACE PHILOSOPHY	WHERE IT'S COVERED ON THIS SITE
Windows 11	GUI-first, with a scriptable CLI (PowerShell) layered alongside — and, as this chapter just named, multiple overlapping GUIs at once	This course, and its own sequel
Linux	CLI-first by tradition; a GUI desktop environment is one optional layer on top, not the default entry point	*Installing and Configuring Linux*, plus the entire Linux/Systems subject
macOS	GUI-first with a genuine Unix core underneath (unlike Windows) — a real third philosophy	Reserved on the bucket list, not yet started

A FIRST PRACTICAL HABIT

Try Snap Layouts right now, on whatever you're reading this on: hover the maximize button of any window, and pick a layout with at least two slots. Windows will prompt you to fill the remaining slot with another open window. This one habit alone is worth more day-to-day time saved than most of what follows in this chapter.

"IT'S JUST WINDOWS 10 WITH ROUNDED CORNERS" IS A COSTLY ASSUMPTION

Treating Windows 11 as a cosmetic refresh means missing real, functional differences — the taskbar's own missing drag-and-drop and vertical-mode support being the clearest example. A support engineer who assumes "if it worked in Windows 10, the same click-path works here" will eventually walk a user through steps that simply don't exist anymore.

Where This Course Is Headed

Installation and first setup, File Explorer and the file system, accounts and Windows Hello, the Settings app in real depth, installing and managing software, networking, updates and maintenance, the built-in security stack, PowerShell basics, accessibility features, and a capstone building a real personal workstation end to end. Windows 11 Troubleshooting & Administration then picks up exactly where this course leaves off — Event Viewer, the Registry, Group Policy, Task Scheduler, and real incident diagnosis — the client-administration skill this whole track was built around.

Hands-On Exercises

EXERCISE 1

On a Windows 11 machine (or from memory of one), find one specific setting that still requires Control Panel rather than the modern Settings app. Explain how you'd confirm whether a setting has fully migrated before assuming Settings alone is enough.

View solution

EXERCISE 2

Use Snap Layouts to arrange three open windows on screen at once. Then explain, in your own words, why this chapter calls the taskbar's own missing vertical-mode and drag-and-drop support a "real functional difference" rather than a cosmetic one.

View solution

EXERCISE 3

This chapter names three interfaces to the same underlying system state — Settings, Control Panel, and (previewed) PowerShell — with a fourth layer still to come in Windows 11 Troubleshooting & Administration. Explain why a support engineer benefits from knowing all of them exist, rather than just the newest one.

View solution

CHAPTER 1 QUICK REFERENCE

- **Start Menu** — centered, pinned apps + Recommended, no more Live Tiles grid
- **Taskbar** — centered icons, no vertical mode, no drag-and-drop onto icons (both existed in Windows 10)
- **Widgets** — a personalized panel that depends on a Microsoft account sign-in
- **Snap Layouts** — hover the maximize button, or `Win + Z`, for instant window-arrangement grids
- **This course's own throughline:** the same system state reachable through Settings, Control Panel, and (later) PowerShell — and, in Course 2, the Registry and Group Policy
- **Next chapter:** Installation & First Setup

CHAPTER 2 OF 12

Installation & First Setup

Windows 11 Fundamentals

Chapter 2 · Installation & First Setup

Chapter 1 toured the interface a running Windows 11 machine presents. This chapter goes one step earlier — the requirements a machine has to meet before Windows 11 will install at all, the installation process itself, and the account decision every fresh install forces on day one. *Build a New Development PC 11* already covered a freshly built machine's first boot and BIOS setup in general terms; this chapter is the OS-side half of that same moment — the specific settings inside that BIOS/UEFI screen Windows 11 actually requires to be enabled.

Hardware Requirements — Stricter Than Any Previous Windows Release

Windows 11 raised its minimum hardware bar meaningfully above Windows 10's — and, unlike most past Windows version bumps, some of the new requirements are about security architecture, not raw performance:

- **TPM 2.0** (Trusted Platform Module) — a dedicated security chip (or firmware-based equivalent) that stores encryption keys outside the reach of the operating system itself. Required, not merely recommended, for a supported install.
- **Secure Boot** — a UEFI firmware feature that only allows cryptographically signed bootloaders to run, blocking a large class of boot-level malware before Windows itself ever starts.
- **A supported 64-bit CPU** — a specific, Microsoft-published list of compatible processor generations, not simply "any 64-bit CPU."
- **4 GB RAM, 64 GB storage minimum** — the raw baseline, on top of everything above.

TPM 2.0 and Secure Boot exist to support Windows 11's own deeper security features — Virtualization-Based Security (VBS) and Hypervisor-Protected Code Integrity (HVCI) among them, both covered from the defensive side in Chapter 9. The hardware requirement isn't arbitrary gatekeeping; it's the foundation those later features are actually built on.

Contrasting With Build a New Development PC 11's Own BIOS/UEFI Material

	WHAT IT COVERS	WHY IT MATTERS HERE
Build a New Development PC 11	General first-boot BIOS/UEFI setup for a freshly assembled machine — boot order, XMP memory profiles, fan curves	Covers the BIOS/UEFI screen itself, in general terms, for any OS
This chapter	The specific toggles inside that same screen — TPM (sometimes labeled "PTT" on Intel or "fTPM" on AMD) and Secure Boot — that Windows 11 specifically requires	A machine can pass every check in Build a New Development PC 11's own setup and still fail Windows 11's installer, if these two toggles are left off

On many motherboards, TPM support exists in firmware already — as Intel PTT (Platform Trust Technology) or AMD fTPM — but ships **disabled by default**, precisely because Windows 10 never required it. A machine built specifically for a fresh Windows 11 install should have both toggles checked and enabled during that same first BIOS/UEFI visit, rather than discovered as an installer error afterward.

Installation Media

Two official routes exist:

- **Windows 11 Installation Assistant** — an in-place upgrade tool for a machine already running a supported Windows version.
- **Media Creation Tool** — downloads an ISO and can write it directly to a bootable USB drive, for a clean install or a new machine with nothing installed yet.

A clean install from bootable USB is the more reliable path for a genuinely new setup — it avoids carrying forward any leftover configuration, drivers, or clutter from a previous install, at the cost of needing to reinstall applications afterward.

The Out-of-Box Experience (OOBE)

First boot after installation walks through region, keyboard layout, a network connection, and — since Windows 11 22H2 — a network connection is required by default before setup will continue on the Home edition, specifically so the account-creation step that follows can push toward a Microsoft account.

Local vs. Microsoft Account — A Real, Ongoing Trade-Off

	GAINS	COSTS
Microsoft account	Settings sync across devices, OneDrive integration, Windows Hello credentials that follow you, easier password recovery	Requires internet during setup, ties the machine's identity to a cloud account, more data shared with Microsoft by default

	GAINS	COSTS
Local account	No cloud dependency, works fully offline, no account-level data leaves the machine	No cross-device sync, password recovery is entirely local (no "forgot password" email flow), Microsoft has made this option progressively harder to find in the OOBE flow over successive updates

CHECKING TPM STATUS ON A RUNNING MACHINE

Run `tpm.msc` from the Run dialog (`Win + R`) on any Windows 11 machine to open the TPM Management console directly — it reports the TPM's presence, version, and status without needing to reboot into the BIOS/UEFI screen at all.

BYPASSING THE REQUIREMENTS HAS REAL, LASTING CONSEQUENCES

Registry-edit and setup-file workarounds to install Windows 11 on unsupported hardware do circulate, and Microsoft's own tolerance of them has shifted from release to release — treat any specific bypass method as temporary, not a documented feature. An install that bypasses the hardware check is explicitly marked unsupported by Microsoft, with no guarantee of continued update delivery. This is a real, practical distinction worth knowing before recommending it to anyone: it may work today and stop working at the next feature update with no warning.

Hands-On Exercises

EXERCISE 1

Explain why TPM 2.0 and Secure Boot are described in this chapter as "the foundation" for Windows 11's deeper security features, rather than security features in their own right.

View solution

EXERCISE 2

A machine passes every check described in Build a New Development PC 11's own first-boot BIOS/UEFI walkthrough, but the Windows 11 installer still refuses to proceed. Using this chapter's own compare-table, explain the most likely cause and how to fix it.

View solution

EXERCISE 3

A user wants to set up a new Windows 11 machine but has no interest in Microsoft account features and wants to avoid handing over an internet connection during setup at all. Using this chapter's own trade-off table, explain what they gain and lose by choosing a local account, and why Microsoft has made that option progressively harder to reach.

📄 View solution

CHAPTER 2 QUICK REFERENCE

- **TPM 2.0 + Secure Boot** — required, and the foundation for VBS/HVCI, covered from the defensive side in Chapter 9
- Often disabled by default in BIOS/UEFI as Intel PTT or AMD fTPM — check and enable both during the same first-boot visit Build a New Development PC 11 already covers
- **Clean install (bootable USB)** vs. **in-place upgrade (Installation Assistant)** — a clean install avoids carried-over clutter
- **tpm.msc** — checks TPM presence/version/status on a running machine without a reboot
- Bypassing hardware requirements is explicitly unsupported and not guaranteed to keep working across updates
- **Next chapter:** File Explorer & the File System

CHAPTER 3 OF 12

File Explorer & the File System

Windows 11 Fundamentals

Chapter 3 · File Explorer & the File System

Chapter 2 left off at account creation — and a Microsoft account, per that chapter's own trade-off table, brings OneDrive integration along with it. This chapter covers the file system Windows actually stores everything on, the File Explorer app most people use to browse it, and how OneDrive sits on top of both without most users ever noticing where the boundary actually is.

NTFS — Windows's Own Default File System

Every modern Windows install formats its system drive as **NTFS** (New Technology File System). A few of its defining features are worth knowing by name:

- **Journaling** — NTFS logs metadata changes before committing them, so an unexpected power loss corrupts, at worst, the file being actively written — not the entire volume.
- **Access Control Lists (ACLs)** — a much finer-grained permission model than a simple owner/group/everyone split: individual users or groups can each be granted a distinct combination of read, write, modify, and execute rights on the same file.
- **Alternate Data Streams (ADS)** — a file can carry hidden secondary data streams alongside its main content, invisible in File Explorer by default. This is also the exact mechanism behind the "unblock" checkbox that appears on files downloaded from the internet — the block itself is stored in a hidden stream, not a visible file property.
- **Built-in compression and quotas** — both configurable per-folder or per-volume, no third-party tooling required.

NTFS vs. ext4 — A First Direct Contrast

Linux Filesystems 2 already covers ext4 — Linux's own traditional default file system — in real depth, including its own journaling modes. Placed side by side, the philosophical difference is genuinely instructive:

	PERMISSION MODEL	CASE SENSITIVITY	JOURNALING
NTFS	ACLs — fine-grained, per-user or per-group, multiple simultaneous rule sets on one file	Case-insensitive by default (case-preserving, but "File.txt" and "file.txt" collide)	Metadata-only by default, matching ext4's own default `data=ordered` mode described in Linux Filesystems 2
ext4	Traditional POSIX owner/group/other, extended with optional ACLs	Case-sensitive — "File.txt" and "file.txt" are two different files	Configurable journaling modes, covered in full in Linux Filesystems 2

NTFS's case-insensitivity is a real, practical difference worth internalizing early — a script or file structure copied from a Linux system where `readme.txt` and `README.txt` genuinely coexist as separate files will silently collapse them into one on a default NTFS volume.

File Explorer — Tabs, Search & the Command Bar

File Explorer gained **tabbed browsing** (Windows 11 22H2 and later) — `Ctrl + T` opens a new tab in the same window, rather than a whole new window per folder, a genuine quality-of-life change for anyone regularly juggling multiple locations at once. The old menu-bar-plus-toolbar layout was also replaced with a simplified command bar, with less-common actions moved behind a "See more" (`...`) button or the right-click context menu.

Search inside File Explorer is powered by the **Windows Search** indexing service, which maintains a background index of file names, and — for supported file types — file contents, so a search returns near-instantly rather than scanning the disk live. Locations outside the default indexed set (a secondary drive, for instance) fall back to a slower, live, non-indexed search unless explicitly added to the index.

OneDrive & Files On-Demand

OneDrive integrates directly into File Explorer as though it were an ordinary local folder — but by default, most files inside it exist as small **placeholder** entries only, not real local copies. This is **Files On-Demand**: opening a placeholder file triggers a download on the spot, and the file's own icon overlay (a cloud, a green checkmark, or a spinning sync arrow) indicates its actual current state.

WHERE A ONEDRIVE FILE "ACTUALLY" LIVES

A cloud-only placeholder occupies almost no local disk space — the visible file size in File Explorer is the file's real size, not the space it currently occupies on disk. Right-click any OneDrive file and check "Free up space" vs. "Always keep on this device" to see and control this state directly, rather than assuming every visible file is fully present locally.

A PRACTICAL HABIT FOR ANYONE DOING OFFLINE WORK

Before traveling or working somewhere without reliable internet, select the specific folders needed and choose "Always keep on this device" — this forces a real local download ahead of time, rather than discovering a placeholder

file that can't be opened once offline.

MOVING FILES OUTSIDE THE ONEDRIVE FOLDER BREAKS SYNC SILENTLY

Dragging a file out of the OneDrive folder tree (to a different drive, for instance) doesn't move it "within OneDrive" — it removes it from OneDrive's own management entirely, with no warning dialog by default. A user who assumes everything on their PC is "backed up to OneDrive" because it once was can lose that protection the moment a file is relocated, without any error appearing at the time.

Hands-On Exercises

EXERCISE 1

A folder copied from a Linux machine contained both "notes.txt" and "Notes.txt" as separate files. After copying it onto an NTFS volume, only one file remains. Using this chapter's own compare-table, explain exactly what happened.

View solution

EXERCISE 2

A user reports that a file inside their OneDrive folder "won't open" while they're on a flight with no internet access, even though the file appeared normally in File Explorer just before takeoff. Explain what's most likely happening, and what step earlier would have prevented it.

View solution

EXERCISE 3

Explain why a search for a file on a secondary, non-default drive can be noticeably slower than the same search on the primary Windows drive, using this chapter's own explanation of how Windows Search actually works.

View solution

CHAPTER 3 QUICK REFERENCE

- **NTFS** — journaling, fine-grained ACLs, hidden Alternate Data Streams, case-insensitive by default
- **ext4** (Linux Filesystems 2) — POSIX permissions, case-sensitive, configurable journaling modes
- **File Explorer tabs** — `Ctrl + T`, added in Windows 11 22H2
- **Windows Search** — indexed locations return near-instant results; non-indexed locations fall back to a slower live scan
- **Files On-Demand** — most OneDrive files are cloud-only placeholders until opened or explicitly pinned locally

- Moving a file out of the OneDrive folder tree silently ends OneDrive's management of it — no warning shown
- **Next chapter:** User Accounts, Permissions & Windows Hello

CHAPTER 4 OF 12

User Accounts, Permissions & Windows Hello

Windows 11 Fundamentals

Chapter 4 · User Accounts, Permissions & Windows Hello

Chapter 2 covered the local-vs-Microsoft account decision made at setup; Chapter 3 previewed NTFS's own fine-grained ACL permission model. This chapter goes one level deeper on both — what an account can actually *do* once signed in, how Windows enforces that boundary moment to moment, and the sign-in methods (PIN, fingerprint, face) that sit on top of whichever account type was chosen.

Administrator vs. Standard User

Every Windows 11 account is either an **administrator** or a **standard user**. An administrator account can install software, change system-wide settings, and modify other users' accounts; a standard user is deliberately restricted from all three. The built-in `Administrator` account itself ships **disabled by default** — the account created during setup is a separate administrator account, not that one.

Running day-to-day as a standard user, with a separate administrator account reserved for the occasions that genuinely need it, is the same **least-privilege** principle *Database Security 3* teaches for database accounts — applied here to a desktop user account instead of a database one. A compromised standard-user session can do meaningfully less damage than a compromised administrator session, for exactly the same underlying reason a database service account with only the permissions it actually needs limits the blast radius of a SQL injection.

User Account Control (UAC) — Enforcing the Boundary in Real Time

UAC is the mechanism that actually enforces the administrator/standard split moment to moment. Even a signed-in administrator runs most processes with standard-user-level rights by default; the moment an action genuinely requires elevation (installing software, changing a protected setting), UAC interrupts with a consent prompt on the **secure desktop** — a separate, isolated desktop session that ordinary applications, including malware already running on the regular desktop, cannot programmatically interact with or dismiss.

WHY THE SECURE DESKTOP SPECIFICALLY MATTERS

Without an isolated secure desktop, malicious software could simply simulate a click on an "Allow" button the instant a prompt appeared, defeating the entire point of asking permission first. The secure desktop closes that exact gap — a UAC prompt genuinely cannot be answered by anything except a real, present user at the keyboard.

UAC's own notification level is adjustable (Control Panel's "Change User Account Control settings," reached from Chapter 1's own legacy-interface material) — from always notifying, down to never notifying at all.

NTFS Permissions in Practice

Chapter 3 introduced NTFS's ACL model in outline; this is what using it actually looks like. Right-clicking any file or folder and opening its **Security** tab shows the exact list of users and groups with explicit permissions on that object, each with their own combination of read, write, modify, and full control. Permissions set on a parent folder are, by default, **inherited** by everything inside it — a folder-level change silently cascades to every file underneath unless inheritance is deliberately broken for a specific subfolder.

PERMISSION	WHAT IT ACTUALLY ALLOWS
Read	View the file's contents and its own listed attributes/permissions
Write	Modify contents, create new files inside a folder
Modify	Read + Write, plus delete
Full Control	Modify, plus change permissions on the object itself and take ownership

Windows Hello — PIN, Fingerprint & Face

Windows Hello covers three sign-in methods: a **PIN**, a **fingerprint** (via a supported reader), and **facial recognition** (via an infrared camera, not a standard webcam — a real hardware distinction, since infrared depth data is what prevents a simple photograph from fooling it). All three rely on the same TPM hardware Chapter 2 already covered as a Windows 11 installation requirement.

A genuinely counter-intuitive but well-documented fact: a Windows Hello PIN is not simply "a shorter, weaker password." It's bound to the specific device via a TPM-backed asymmetric key pair — the PIN itself never leaves the machine and is never transmitted anywhere, unlike a password, which is what actually gets sent to (and potentially phished or reused across) a remote service. A leaked PIN from one device is useless on any other device, since there's no matching private key anywhere else to pair it with.

CHECKING YOUR OWN ACCOUNT TYPE

Settings > Accounts > Your info shows the current account's type directly beneath the account name — "Administrator" or nothing shown at all for a standard account. No elevated prompt or additional tool is needed just to check.

TURNING UAC ALL THE WAY OFF REMOVES REAL PROTECTION, NOT JUST A NAG SCREEN

Setting the UAC slider to "Never notify" doesn't just silence prompts — every process, including malicious ones, then runs with full administrator rights by default, with nothing left to interrupt a silent privilege escalation. Treat "UAC is

annoying" as a prompt to understand what's triggering it, not a reason to disable the mechanism itself.

Hands-On Exercises

EXERCISE 1

Explain, using this chapter's own comparison, why running day-to-day as a standard user with a separate administrator account for occasional use mirrors the same principle Database Security 3 teaches for database accounts.

View solution

EXERCISE 2

Explain why a UAC consent prompt appears on a separate secure desktop rather than as an ordinary on-screen dialog, and what specific attack this design choice prevents.

View solution

EXERCISE 3

A colleague argues that a 4-digit Windows Hello PIN is objectively weaker than their own long, complex account password, and should be disabled. Using this chapter's own explanation, explain why that reasoning misses a key structural difference between the two.

View solution

CHAPTER 4 QUICK REFERENCE

- **Administrator vs. standard user** — the built-in Administrator account is disabled by default; the setup-created admin account is a separate one
- Least privilege for desktop accounts mirrors Database Security 3's own principle for database accounts
- **UAC** — elevation prompts run on an isolated secure desktop malware cannot programmatically click through
- **NTFS Security tab** — Read/Write/Modify/Full Control, inherited from parent folders by default
- **Windows Hello PIN** — TPM-bound and device-specific, not a "shorter password," and not transmittable like one
- Turning UAC to "Never notify" removes real protection, not just prompt fatigue
- **Next chapter:** The Settings App, In Depth

CHAPTER 5 OF 12

The Settings App, In Depth

Windows 11 Fundamentals

Chapter 5 · The Settings App, In Depth

Chapter 1 named Settings as the first of this course's own three (soon four) interfaces to the same underlying system state. Chapter 4 already used two of its pages directly — account type and UAC's own notification slider. This chapter is the real tour: what each major Settings category actually covers, and exactly where the still-incomplete migration away from Control Panel stands today.

System

The largest category, covering Display (resolution, scaling, HDR, multiple-monitor arrangement), Sound (output/input devices, volume mixer per-app), Notifications & actions, Focus assist (temporarily silencing notifications), Power & battery, and Storage (a high-level usage breakdown, with Storage Sense's own automatic cleanup covered in full in Chapter 8). Most of what a typical user adjusts day to day lives somewhere in this one category.

Apps

Installed apps (a searchable, sortable list with per-app uninstall and "modify" options), Optional features (component-level extras like language packs or legacy protocol support, installed separately from ordinary apps), Default apps (which program opens a given file type or protocol by default), and Startup apps (what launches automatically at sign-in, each entry showing an actual measured startup-impact rating). Installing new software itself — the Microsoft Store, `winget`, and traditional installers — is covered in full in Chapter 6.

Personalization

Background and themes, colors (system-wide light/dark mode plus a customizable accent color, applied consistently across Settings, File Explorer, and most built-in apps), fonts, and — directly relevant after Chapter 1's own interface tour — Taskbar personalization, including the setting that moves centered icons back to the classic left-aligned position.

Privacy & Security

Per-permission toggles for camera, microphone, location, and other sensitive capabilities — each with both a system-wide switch and a separate per-app list underneath it. This category also hosts the diagnostic data level (how much telemetry is sent to Microsoft) and is the entry point into the Windows Security app, covered in full in Chapter 9.

> **A GLOBAL PRIVACY TOGGLE SILENTLY OVERRIDES EVERY PER-APP ONE**
>
> Turning off "Microphone access" at the system-wide level disables it for every app, regardless of whatever each individual app's own toggle is set to underneath. A user who reports "my microphone stopped working in one specific app" after previously working fine is often looking at the global switch, not the per-app one — checking only the per-app list first is a common, easy-to-repeat troubleshooting miss.

The Ongoing Control Panel Migration, Concretely

Chapter 1 named this course's own throughline in the abstract; here's what it looks like in practice today:

AREA	MIGRATION STATUS
Display, Sound, Personalization, Accounts	Fully migrated — Control Panel equivalents either redirect straight into Settings or no longer exist
Network adapters, some printer management	Partially migrated — a basic view exists in Settings, but advanced configuration still opens the underlying Control Panel dialog
Advanced user/group management, some legacy hardware configuration	Not migrated at all — Control Panel (or a dedicated legacy tool like Local Users and Groups) remains the only interface

Windows 11 Troubleshooting & Administration 1 goes further still, adding the Registry and Group Policy as a deeper administrative layer beneath both Settings and Control Panel — the point where a support engineer stops relying on either GUI at all for certain classes of problem.

> **DON'T HUNT THROUGH CATEGORIES — SEARCH**
>
> The search bar at the top of the Settings app indexes every setting across every category, including many that are nested several pages deep. Typing a keyword and pressing Enter is consistently faster than navigating category by category, especially for a setting whose exact home isn't obvious in advance.

Hands-On Exercises

EXERCISE 1

A user says a specific app "lost permission" to use the microphone overnight with no changes made inside that app itself. Using this chapter's own warn-box, explain the most likely cause and where to check first.

View solution

EXERCISE 2

Using this chapter's own migration table, explain why "just use Settings for everything" isn't yet reliable advice for every kind of Windows 11 configuration task.

View solution

EXERCISE 3

Explain why this chapter recommends using the Settings app's own search bar rather than navigating category by category, and what kind of setting benefits from that habit the most.

View solution

CHAPTER 5 QUICK REFERENCE

- **System** — Display, Sound, Notifications, Focus assist, Power & battery, Storage
- **Apps** — Installed apps, Optional features, Default apps, Startup apps
- **Personalization** — Background, Colors (light/dark + accent), Taskbar layout
- **Privacy & Security** — per-app and global permission toggles, diagnostic data, Windows Security entry point
- A global permission toggle silently overrides every per-app toggle underneath it
- The Control-Panel-to-Settings migration is real but incomplete — some areas are fully migrated, some partially, some not at all
- **Next chapter:** Installing & Managing Software

CHAPTER 6 OF 12

Installing & Managing Software

Windows 11 Fundamentals

Chapter 6 · Installing & Managing Software

Chapter 5 introduced the Apps category's own Installed apps, Optional features, and Default apps pages without going deep on any of them. This chapter is that depth — three genuinely different ways software actually arrives on a Windows 11 machine, how default file-type associations really work, and how to remove software cleanly rather than just "delete the folder and hope."

The Microsoft Store — Sandboxed by Design

Store apps are packaged as **MSIX**, a format built around a real sandbox — each app runs with its own isolated storage and a specific, declared set of permissions, rather than the broad, largely unrestricted system access a traditional installer's own program gets by default. Store apps also update automatically in the background, and uninstall cleanly by design: no leftover registry entries or orphaned files, since the sandbox boundary that limited the app's reach while running also bounds exactly what has to be removed.

winget — Windows's Own Package Manager

winget (the Windows Package Manager) installs, upgrades, and removes software entirely from the command line — previewed here, with PowerShell itself covered in Chapter 10. It's a genuinely useful point of comparison against *Installing and Configuring Linux 10*'s own package-manager coverage: both solve the same underlying problem (find a package by name, fetch it, install it, track what's installed for later updates), though winget's own catalog is a curated list of installer/MSIX manifests rather than a single distribution-maintained repository the way a Linux package manager's own repos work.

```
# Search for a package by name
winget search "visual studio code"

# Install a specific package
winget install Microsoft.VisualStudioCode

# List every installed package winget can see
winget list

# Upgrade everything winget manages, at once
winget upgrade --all
```

Traditional Installers — MSI and EXE

.msi packages are run through the Windows Installer service, a standardized format that provides built-in rollback (an interrupted or failed install cleanly reverts) and a genuine repair option (reinstalling only what's missing or corrupted, without a full fresh install). Plain **.exe** installers, by contrast, are just ordinary programs that happen to install something — there's no format-level guarantee of a clean rollback or a standard, predictable uninstall behavior; whatever the installer's own author built in is all there is.

	SANDBOXING	UPDATE MECHANISM	UNINSTALL GUARANTEE
Microsoft Store (MSIX)	Full sandbox, declared permissions	Automatic, background	Clean by design — no leftovers
winget	Depends on what it's installing (MSIX or a traditional installer underneath)	Manual, via `winget upgrade`	Depends on the underlying package type
MSI	None — standard system access	Manual	Format-guaranteed rollback/repair, but not sandboxed removal
EXE	None — standard system access	Manual, or self-updating if the app builds it in	Entirely up to the installer's own author

Default Apps — File-Type & Protocol Associations

Settings > Apps > Default apps controls which program opens a given file type (`.pdf` , `.jpg`) or protocol (`http://` , `mailto:`) by default. Windows 11 made a deliberate, publicly documented change here from Windows 10: there's no longer a single "Set as default browser" button that reassigns every relevant file type and protocol at once. Each file type and each protocol must be set individually, one at a time.

> **"SET AS DEFAULT" DOESN'T DO WHAT IT USED TO**
>
> A user switching browsers who expects one click to fully replace their previous default will find Windows 11 nudging them toward setting `.htm` , `.html` , `http` , and `https` individually — a real, deliberate friction point (widely reported as a response to antitrust scrutiny over steering users back toward Edge), not a bug or a missing feature the user simply hasn't found yet.

Clean Uninstalls

Settings > Apps > Installed apps provides uninstall for most software, regardless of how it was originally installed. Traditional MSI/EXE installers, per the format distinction above, don't universally guarantee every file and registry entry is removed — some leftover data is common and, for most users, harmless clutter rather than a real problem. Chasing down every last leftover registry key by hand is exactly the kind of task Windows

11 Troubleshooting & Administration's own Registry chapter is built for, once real diagnostic need (not just tidiness) calls for it.

A FAST SOFTWARE AUDIT

`winget list` gives a quick, complete inventory of installed software winget can see, in one command — useful for auditing a machine's software before troubleshooting, without opening Settings and scrolling through the Installed apps list by hand.

Hands-On Exercises

EXERCISE 1

Explain why a Microsoft Store app uninstalls without leaving behind leftover files or registry entries, while a traditional EXE installer offers no such guarantee.

View solution

EXERCISE 2

A user just switched their default browser and is frustrated that some links still open in their old browser. Using this chapter's own explanation, describe what's actually happening and what they need to do differently.

View solution

EXERCISE 3

Explain how winget's own approach to installing software is similar to, and different from, what Installing and Configuring Linux 10 covers for a Linux package manager.

View solution

CHAPTER 6 QUICK REFERENCE

- **Microsoft Store (MSIX)** — sandboxed, auto-updating, clean uninstall by design
- **winget** — command-line package management, previewing Chapter 10's own PowerShell coverage
- **MSI** — format-guaranteed rollback/repair; **EXE** — no such guarantee, entirely author-dependent
- Windows 11 requires setting default apps per file type/protocol — no single "set as default browser" button, unlike Windows 10
- Leftover files/registry entries from traditional installers are common and usually harmless — real cleanup is a Course 2 topic

- **Next chapter:** Networking on Windows 11

CHAPTER 7 OF 12

Networking on Windows 11

Windows 11 Fundamentals

Chapter 7 · Networking on Windows 11

Networking Fundamentals already covers the underlying concepts — IP addressing, DNS, ports, the whole stack — in real depth, independent of any one operating system. This chapter is the Windows-specific half: how those concepts surface in Windows 11's own settings, and the handful of Windows-only behaviors (network profiles chief among them) that don't exist in the general model at all.

Network Profiles — Public vs. Private

Every network Windows connects to is assigned a **profile** — **Private** (a trusted home or work network) or **Public** (an untrusted network, like a coffee shop or airport). The profile isn't cosmetic labeling; it changes real, active behavior:

	NETWORK DISCOVERY	FILE/PRINTER SHARING	FIREWALL RULE SET
Private	On — other devices on the network can see this PC	Enabled by default	Looser, trusted-network rule set
Public	Off — this PC stays invisible to other devices	Disabled by default	Stricter, untrusted-network rule set

This is enforced by **Windows Firewall** maintaining genuinely separate rule sets per profile — a rule enabled for Private doesn't apply while connected to a network currently classified as Public, and vice versa, without needing to be manually toggled each time.

PICKING THE WRONG PROFILE HAS REAL, SILENT CONSEQUENCES

Selecting Public on a genuinely trusted home network (or Private on an untrusted one) doesn't produce an error — it silently applies the wrong rule set. A user who can't find a shared printer, or can't see another PC on their own home network, has very often connected under the Public profile rather than Private, with nothing in the interface actively flagging the mismatch.

Wi-Fi & Ethernet Setup

Wi-Fi connects through the taskbar's quick-settings panel or Settings > Network & internet > Wi-Fi, with saved networks reconnecting automatically by signal strength and priority. A Wi-Fi connection can also be marked **metered** — signaling to Windows that data usage should be minimized, which, notably, can pause or

defer large background downloads including Windows Update itself (covered fully in Chapter 8). Ethernet, by contrast, is normally zero-configuration — DHCP hands out an IP address, gateway, and DNS servers automatically the moment a cable is plugged in.

Manually configuring a static IP, gateway, or DNS server still opens the classic adapter properties dialog — the exact "partially migrated" case Chapter 5's own table already named: Settings shows a basic view, but this deeper configuration remains Control Panel territory underneath.

DNS on Windows — the Practical Side

Networking Fundamentals 9 already covers what DNS actually is and how resolution works, in full. On a Windows machine specifically, DNS servers are set per-adapter (inherited from DHCP by default, or overridden manually in the same adapter properties dialog above), and a local DNS cache is maintained to avoid re-querying for names already resolved recently — the exact cache Windows 11 Troubleshooting & Administration's own networking chapter clears with `ipconfig /flushdns` when a stale entry is the actual problem.

Sharing — Network Discovery & File Sharing

The classic **Network and Sharing Center** Control Panel item remains fully alive and is still where several sharing-related toggles genuinely live, alongside their Settings-app counterparts. Windows's old "Homegroup" feature was removed entirely in favor of simpler folder- and printer-level sharing, controlled per-object through each item's own Properties > Sharing tab, gated by whichever network profile — Private or Public — is currently active.

Windows Firewall Basics

Windows Firewall filters both **inbound** connections (something outside trying to reach this PC) and **outbound** connections (this PC trying to reach something outside), with separate rule sets for the Domain, Private, and Public profiles. Most day-to-day use never requires touching a firewall rule directly — installed apps typically register their own rules automatically — but the Windows Security app (covered in full in Chapter 9) is the modern entry point when a rule genuinely does need reviewing or adding.

> **CHECKING THE CURRENT NETWORK'S PROFILE**
>
> Settings > Network & internet > [the active connection] shows "Network profile type" directly, with a toggle right there to switch between Public and Private — worth checking any time sharing or discovery behaves unexpectedly, before assuming something deeper is broken.

Hands-On Exercises

EXERCISE 1

A user at home can't see their own PC listed on another device connected to the same Wi-Fi network. Using this chapter's own compare-table, explain the most likely cause and where to check it.

View solution

EXERCISE 2

Explain why manually setting a static DNS server on a Windows 11 machine still requires opening the classic adapter properties dialog, tying your answer back to Chapter 5's own migration table.

View solution

EXERCISE 3

Explain why marking a Wi-Fi connection as "metered" can cause a Windows Update to silently stop downloading, and why this isn't a bug.

View solution

CHAPTER 7 QUICK REFERENCE

- **Network profile (Public/Private)** — actively changes discovery, sharing, and firewall rule sets, not just a label
- Wrong profile selection fails silently — no error, just the wrong rule set applied
- **Metered connections** — can pause background downloads including Windows Update (Chapter 8)
- Static IP/DNS configuration still opens the classic adapter properties dialog — Chapter 5's own "partially migrated" case
- **Networking Fundamentals 9** covers DNS concepts in full; this chapter covers the Windows-specific practical side
- Windows Firewall maintains separate rule sets per profile automatically
- **Next chapter:** Windows Update & Everyday Maintenance

CHAPTER 8 OF 12

Windows Update & Everyday Maintenance

Windows 11 Fundamentals

Chapter 8 · Windows Update & Everyday Maintenance

Chapter 7 previewed one specific Windows Update behavior — deferring on a metered connection — without explaining the update system itself. This chapter covers that system properly, plus the two maintenance tools (Storage Sense and Disk Cleanup) that quietly keep a drive from filling up, one of which reaches directly back into Chapter 3's own OneDrive material.

Feature Updates vs. Quality Updates

	WHAT IT CONTAINS	FREQUENCY	SIZE/IMPACT
Quality update	Security patches, bug fixes — cumulative, each one includes every previous fix	Monthly ("Patch Tuesday")	Smaller, faster install, usually one reboot
Feature update	A genuine version upgrade — new capabilities, UI changes, sometimes new minimum requirements	Roughly annual (e.g. 22H2, 23H2, 24H2)	Much larger, effectively a mini-reinstall of core components

Chapter 7's own metered-connection behavior applies most visibly to quality updates, since they're the ones downloading silently and often in the background; a feature update is large enough that Windows typically prompts explicitly before installing it, rather than pushing it through unannounced the way a routine quality update can.

Deferral & Pausing

Settings > Windows Update > Advanced options allows pausing updates entirely for up to five weeks — useful before an important presentation or demo where an unexpected reboot would be genuinely disruptive — plus "Active hours," a window during which Windows won't automatically restart to finish installing an update already downloaded. Organizations manage this at scale through Windows Update for Business policies, deferring feature and quality updates independently by a configured number of days — a topic that belongs to real IT administration rather than this course's own individual-workstation scope.

Storage Sense

Storage Sense (Settings > System > Storage) automatically frees disk space on a schedule: temporary files, items sitting in the Recycle Bin past a configurable age, and downloads folder items past a configurable age. Its most genuinely OneDrive-aware behavior, though, is this:

STORAGE SENSE CAN SILENTLY REVERSE A CHAPTER 3 DECISION

Storage Sense can automatically convert locally available OneDrive files back into cloud-only placeholders if they haven't been opened recently — exactly the placeholder state Chapter 3 covered as Files On-Demand's own default behavior. A file explicitly downloaded, or even one manually marked "Always keep on this device," can still be affected unless that setting is genuinely respected by the configured Storage Sense policy — worth checking directly if a previously-available file mysteriously needs an internet connection to open again.

Disk Cleanup — the Legacy Tool That's Still Genuinely Useful

`cleanmgr.exe` predates Storage Sense by well over a decade and still exists, unmigrated, as one more instance of Chapter 5's own "not migrated at all" category. It remains the most direct way to reach two things Storage Sense doesn't cover at all: **system file cleanup** (old Windows Update installation files, previous-version rollback data) and the **WinSxS component store** (the side-by-side assembly cache backing Windows's own servicing and update model) — both genuinely capable of reclaiming several gigabytes on a machine that's been through a few feature updates.

CHECKING UPDATE HISTORY BEFORE ASSUMING SOMETHING IS BROKEN

Settings > Windows Update > Update history lists every update actually installed, with success/failure status for each — a fast first check when a machine seems to be behaving differently after "something updated," rather than guessing at which update might be responsible.

A LARGE FEATURE UPDATE NEEDS REAL FREE SPACE FIRST

A feature update effectively stages a near-complete secondary copy of core Windows components before switching over, and can fail partway through — or refuse to start at all — on a drive that's nearly full. Running Disk Cleanup's own system file cleanup or reviewing Storage Sense's own recent activity before a feature update is due is a genuinely useful habit, not just general housekeeping.

Hands-On Exercises

EXERCISE 1

Using this chapter's own compare-table, explain why a quality update is far more likely to install silently in the background than a feature update, and why that difference matters for someone on a metered connection per Chapter 7.

View solution

EXERCISE 2

A user marked an important file "Always keep on this device" in Chapter 3, but months later finds it needs to download again before opening. Using this chapter's own finding-box, explain what most likely happened.

View solution

EXERCISE 3

Explain why Disk Cleanup, a tool that predates Storage Sense by over a decade, is still genuinely useful today rather than fully superseded.

View solution

CHAPTER 8 QUICK REFERENCE

- **Quality updates** — monthly, cumulative, security/bug fixes; **feature updates** — roughly annual, much larger, a real version upgrade
- Pausing updates (up to 5 weeks) and Active hours both exist for genuinely disruptive-timing scenarios
- **Storage Sense** can silently convert a locally available OneDrive file back to a cloud-only placeholder — revisiting Chapter 3's own Files On-Demand mechanism
- **Disk Cleanup** — legacy but still uniquely useful for system files and the WinSxS component store
- Free space matters before a feature update — it stages a near-complete secondary copy first
- **Next chapter:** Built-in Security Features

CHAPTER 9 OF 12

Built-in Security Features

Windows 11 Fundamentals

Chapter 9 · Built-in Security Features

Chapters 5 and 7 both pointed toward the Windows Security app without opening it. This chapter finally does — Defender, BitLocker, and Smart App Control, plus the TPM and UAC material from Chapters 2 and 4 finally paying off as the real foundation two of these features are built directly on top of.

The Windows Security App — One Dashboard, Several Engines

Windows Security consolidates several genuinely separate protection systems into one interface: Virus & threat protection, Account protection (where Windows Hello, covered in Chapter 4, actually lives), Firewall & network protection (Chapter 7's own firewall material, surfaced here), App & browser control (home to Smart App Control), and Device security (TPM status and, on capable hardware, BitLocker/Device Encryption).

Microsoft Defender Antivirus

Defender is built in and active by default, combining real-time protection (scanning files as they're accessed) with cloud-delivered protection (checking suspicious files against Microsoft's own up-to-the-minute threat intelligence rather than relying solely on a locally stored signature file). Installing a third-party antivirus product automatically disables Defender's own real-time scanning to avoid conflicts — but Defender still performs periodic background scans even then, as a genuine second layer rather than switching off entirely.

BitLocker — Full-Disk Encryption, TPM-Sealed

BitLocker encrypts an entire drive, decrypting it transparently the moment Windows boots successfully and the TPM — the same chip Chapter 2 covered as a Windows 11 installation requirement — releases the encryption key. If the drive is removed and connected to a different machine, or if the boot process is tampered with, the TPM refuses to release the key at all, and the drive remains fully encrypted and unreadable.

On many Home-edition machines meeting the hardware requirements, a lighter version called **Device Encryption** turns on automatically the moment a Microsoft account signs in — a real, easy-to-miss

consequence of Chapter 2's own account-type decision, tying that setup-time choice directly to whether disk encryption is silently active today.

BitLocker vs. Database Security 5's Own Encryption-at-Rest Material

	WHAT IT ACTUALLY PROTECTS AGAINST	WHEN THE DATA IS DECRYPTED
BitLocker	Physical theft of the device or drive — the whole volume is unreadable without the TPM-released key	Transparently, the moment Windows itself boots successfully on the original hardware
Database TDE (Database Security 5)	Direct filesystem/backup-file access bypassing the database engine entirely	Transparently, the moment the database engine itself opens the file with its own key

Both are genuinely "encryption at rest," and both decrypt transparently for the legitimate process that's supposed to have access — but they protect against a different specific threat, at a different layer entirely, exactly the kind of same-name-different-mechanism distinction worth keeping straight rather than assuming one implies the other.

Key Management — a Real Instance of Cryptography Fundamentals 11

BitLocker's own recovery key — a 48-digit backup, generated once and requiring the user to save it somewhere outside the encrypted drive itself (a Microsoft account, a USB drive, printed, or Active Directory in a managed environment) — is a direct, practical instance of the key-storage and backup discipline *Cryptography Fundamentals 11* covers in general terms. There is no backdoor: losing both the TPM-sealed key and the recovery key means the data is genuinely, permanently unrecoverable.

Smart App Control

Smart App Control (Windows 11 22H2 and later) uses cloud-based AI and code-signing verification to block unknown or unsigned applications from running at all, before they get the chance to do anything — a meaningfully more proactive stance than Defender's own scan-after-the-fact model.

SMART APP CONTROL IS A ONE-WAY TOGGLE

Smart App Control can only be turned on cleanly on a fresh Windows 11 install (or left on if it was already active). Once disabled, it cannot be turned back on without a clean reinstall — a genuinely unusual, one-directional setting worth knowing about before disabling it casually, since it's not a mistake that a later Settings visit can simply undo.

CHECKING BITLOCKER STATUS WITHOUT OPENING THE FULL APP

Settings > Privacy & security > Device encryption (or the Device Security page's own BitLocker section on Pro editions) shows current encryption status directly — worth checking on any machine handling sensitive data, since Device Encryption's automatic activation, per this chapter, can already be silently active without anyone having deliberately turned it on.

Hands-On Exercises

EXERCISE 1

Explain why removing a BitLocker-encrypted drive and connecting it to a different machine doesn't allow that data to be read, tying your answer back to Chapter 2's own TPM material.

View solution

EXERCISE 2

Using this chapter's own compare-table, explain why BitLocker and a database's own Transparent Data Encryption are both genuinely "encryption at rest," yet protecting against them being interchangeable would be a mistake.

View solution

EXERCISE 3

A colleague wants to quickly test disabling Smart App Control "just to see what changes," planning to turn it back on immediately afterward. Explain, using this chapter's own warn-box, why that plan won't work as expected.

View solution

CHAPTER 9 QUICK REFERENCE

- **Windows Security** — one dashboard covering Defender, Account protection (Windows Hello), Firewall, App & browser control, Device security
- **BitLocker** — full-disk encryption, TPM-sealed, protects against physical theft; recovery key is a real Cryptography Fundamentals 11-style key-management scenario
- **Device Encryption** — the lighter Home-edition version, can activate silently just by signing in with a Microsoft account
- BitLocker and database TDE are both "encryption at rest" but protect against genuinely different threats
- **Smart App Control** — proactive, blocks unsigned apps before execution; a one-way toggle once disabled
- **Next chapter:** PowerShell Basics for Windows Users

CHAPTER 10 OF 12

PowerShell Basics for Windows Users

Windows 11 Fundamentals

Chapter 10 · PowerShell Basics for Windows Users

Chapter 1 named PowerShell as the third of this course's own three interfaces to the same underlying system state. Chapter 6 used it in passing for `winget`; Chapter 7 previewed `ipconfig /flushdns`. This chapter is PowerShell in its own right — how it actually works, and why it's a genuinely different kind of shell from anything *Bash Scripting Fundamentals* already covers, not just a Windows-flavored equivalent.

PowerShell vs. cmd — Two Very Different Shells Sharing a Terminal

`cmd.exe` is Windows's original command interpreter, tracing back to MS-DOS — still present, still occasionally necessary for a handful of legacy batch scripts, but with no real scripting language behind it. **PowerShell** is a genuine scripting environment with variables, functions, and real data types. A further wrinkle worth knowing: **Windows PowerShell 5.1** ships built into Windows itself, while **PowerShell 7+** is a separate, cross-platform, open-source install — Windows Terminal defaults to 5.1 unless 7 has been explicitly installed and set as the default profile, a genuinely common source of "which PowerShell am I even using" confusion.

The Object Pipeline — PowerShell's Real Defining Difference

This is the single most important contrast with *Bash Scripting Fundamentals*'s own coverage. Bash pipes plain **text** between commands — each command receives a stream of characters and has to parse it back into meaningful data itself, often with tools like `awk` or `sed` doing that parsing. PowerShell pipes **objects** — structured data with real properties — so a command receiving piped input already has direct access to named fields, with no text-parsing step required at all.

```
# List every running process using more than 200MB of memory
Get-Process | Where-Object { $_.WorkingSet -gt 200MB }

# Sort those results by memory usage, descending
Get-Process | Where-Object { $_.WorkingSet -gt 200MB } | Sort-Object WorkingSet -Descending
```

`$_.WorkingSet` refers directly to a named property on the process object flowing through the pipeline — no splitting a line of text on whitespace, no guessing which column holds the memory figure, the way an equivalent Bash pipeline built around `ps` and `awk` would need to.

	PIPELINE DATA	COMMAND NAMING	PLATFORM
PowerShell	Structured objects with real properties	Consistent Verb-Noun cmdlets (`Get-Process`, `Stop-Service`)	Windows built-in (5.1); PowerShell 7+ is cross-platform
Bash	Plain text streams	A large set of independently named Unix utilities, each with its own conventions	Linux/macOS native; available on Windows via WSL

Cmdlets & Discovery — Get-Help, Get-Command

Every built-in PowerShell command is a **cmdlet**, consistently named `Verb-Noun` — `Get-Process`, `Stop-Service`, `Set-ExecutionPolicy`. `Get-Help <cmdlet> -Examples` shows real usage examples for any cmdlet directly in the terminal; `Get-Command *service*` finds every cmdlet whose name contains a given word, a genuinely fast way to discover a command whose exact name isn't already known.

A First Script — and the Execution Policy Gotcha

A PowerShell script is simply a text file saved with a `.ps1` extension. By default, though, Windows blocks scripts from running at all — a security measure called **Execution Policy**, distinct from anything covered so far in this course.

```
# A simple first script — free disk space on the C: drive, in GB
Get-PSDrive C | Select-Object @{Name="FreeGB";Expression={[math]::Round($_.Free / 1GB, 2)}}
```

"RUNNING SCRIPTS IS DISABLED ON THIS SYSTEM"

A script downloaded from the internet and run directly will typically fail with this exact error — the default Execution Policy (`Restricted`) blocks all scripts, downloaded or otherwise. The fix, `Set-ExecutionPolicy RemoteSigned`, allows locally written scripts to run freely while still requiring downloaded scripts to be digitally signed — a real, deliberate middle ground, not something to bypass entirely with `Unrestricted` just to make an error go away.

TAB COMPLETION IS A REAL DISCOVERY TOOL, NOT JUST A SHORTCUT

Typing a partial cmdlet name and pressing Tab cycles through every matching cmdlet PowerShell knows about — genuinely useful for recalling a command's exact name or spelling without leaving the terminal to search for it.

Hands-On Exercises

EXERCISE 1

Explain why a PowerShell pipeline built around `Get-Process | Where-Object` doesn't need to parse text the way an equivalent Bash pipeline built around `ps` and `awk` would.

View solution

EXERCISE 2

A user downloads a .ps1 script and double-clicks it, but nothing happens — running it from a terminal instead shows "running scripts is disabled on this system." Explain what's happening and the recommended fix, using this chapter's own warn-box.

View solution

EXERCISE 3

Explain why "Windows PowerShell" and "PowerShell 7" being two genuinely different things, rather than just two version numbers of the same product, is worth knowing before troubleshooting a script that behaves differently on two machines.

View solution

CHAPTER 10 QUICK REFERENCE

- **cmd.exe** — legacy, no real scripting language; **PowerShell** — genuine scripting environment
- **Windows PowerShell 5.1** (built in) vs. **PowerShell 7+** (separate, cross-platform install) — two genuinely different products
- PowerShell's object pipeline vs. Bash's text pipeline is the single biggest structural difference between the two shells
- **Verb-Noun cmdlets**, discovered via `Get-Help` and `Get-Command`
- **Execution Policy** blocks scripts by default; `RemoteSigned` is the real recommended middle ground, not `Unrestricted`
- **Next chapter:** Accessibility & Productivity Features

CHAPTER 11 OF 12

Accessibility & Productivity Features

Windows 11 Fundamentals

Chapter 11 · Accessibility & Productivity Features

Chapter 1 covered Snap Layouts as one multitasking tool; Chapter 4 covered the secure desktop as an isolated system-level session. This chapter rounds out both threads — the built-in accessibility suite, a second multitasking tool that works alongside Snap Layouts rather than replacing it, and two smaller productivity features worth building into daily habit.

Narrator — Built-in Screen Reading, Even Before Sign-In

Narrator (`Win + Ctrl + Enter`) reads screen content aloud, navigable by keyboard alone. A genuinely notable detail: Narrator can be invoked directly from the **sign-in screen itself**, before any user has authenticated at all — this works because the sign-in screen runs in its own isolated system session, conceptually the same kind of separation Chapter 4 covered for UAC's own secure desktop. Just as a UAC prompt runs somewhere ordinary desktop applications can't reach, the sign-in screen's own accessibility tools run in a session available regardless of which account (if any) ends up signing in.

Magnifier & Voice Access

Magnifier (`Win + Plus` to open, `Win + Minus` to zoom out, `Win + Esc` to close) offers three modes: Full screen, Lens (a movable magnified rectangle following the cursor), and Docked (a fixed magnified strip along one edge). **Voice access** allows controlling the entire system — clicking, typing, dictating — by voice alone, and requires a one-time download of its speech recognition model before first use.

Virtual Desktops — Task View

`Win + Tab` opens **Task View**, which manages entirely separate virtual desktops — each with its own independent set of open windows. This is a genuinely different multitasking tool from Chapter 1's own Snap Layouts, not a competing one: Snap Layouts arranges multiple windows within one desktop at once, while virtual desktops separate different tasks (say, focused work vs. communication apps) onto entirely different desktops switched between with `Ctrl + Win + Left/Right`, each one able to use its own Snap Layout arrangement independently.

Clipboard History

`Win + V` opens clipboard history — retaining multiple recently copied items rather than just the single most recent one, with the option to pin frequently reused entries permanently. Clipboard history is disabled by default and, once enabled, can optionally sync across devices signed into the same Microsoft account — the same account-level sync mechanism Chapter 2's own trade-off table already named as a genuine benefit of choosing a Microsoft account over a local one.

CLIPBOARD HISTORY ISN'T A PERMANENT, SECURE STORE

Sensitive data — passwords, personal information — copied while clipboard history is enabled remains listed there until manually cleared, and syncing it across devices means that data travels to every other signed-in device too. Clearing clipboard history after copying anything sensitive is a real, worthwhile habit, not excessive caution.

Snipping Tool

`Win + Shift + S` opens the Snipping Tool directly into capture mode — rectangular, window, full-screen, or freeform — with a delayed-capture option for grabbing menus that disappear when clicked away from. The modern Snipping Tool also merges in the older Snip & Sketch app's own basic annotation features (drawing, highlighting, cropping) directly on a captured image before saving or sharing it.

A QUICK SHORTCUT REFERENCE

- `Win + Ctrl + Enter` — Narrator
- `Win + Plus` / `Win + Esc` — Magnifier open/close
- `Win + Tab` — Task View (virtual desktops)
- `Win + V` — Clipboard history
- `Win + Shift + S` — Snipping Tool capture

Hands-On Exercises

EXERCISE 1

Explain why Narrator being usable at the sign-in screen, before any account has authenticated, is conceptually related to what Chapter 4 covered about UAC's own secure desktop.

View solution

EXERCISE 2

Explain why virtual desktops (Task View) and Snap Layouts from Chapter 1 are described as working alongside each other rather than one replacing the other.

View solution

EXERCISE 3

A user copies a password to paste into a form, then later shows a colleague their clipboard history for something unrelated — and the password is still listed. Explain why, using this chapter's own warn-box, and what habit would have prevented it.

View solution

CHAPTER 11 QUICK REFERENCE

- **Narrator** — available even at the sign-in screen, conceptually related to Chapter 4's own secure desktop isolation
- **Magnifier** — Full screen, Lens, and Docked modes; **Voice access** — full voice control, needs a one-time model download
- **Task View / virtual desktops** — separates tasks across desktops, complementing rather than replacing Chapter 1's own Snap Layouts
- **Clipboard history** — off by default, can sync via a Microsoft account per Chapter 2, not a secure permanent store
- **Snipping Tool** — rectangular/window/full-screen/freeform capture with built-in annotation
- **Next chapter:** Capstone — Setting Up a Personal Windows 11 Workstation

CHAPTER 12 OF 12

Capstone: Setting Up a Personal Windows 11 Workstation

Windows 11 Fundamentals

Chapter 12 · Capstone: Setting Up a Personal Windows 11 Workstation

Eleven chapters, one system. This capstone walks a single, real workstation setup end to end — a fresh Windows 11 install for a hypothetical freelance developer, "Priya," setting up a new personal machine — deliberately touching every prior chapter in the order a genuine setup would actually hit them, rather than revisiting each one in isolation.

The Scenario

Priya has a newly built PC (fresh from *Build a New Development PC 11*'s own first-boot BIOS visit) and wants it fully set up for daily development work, with reasonable security, without spending an entire weekend on it.

Step 1 — Confirming the Machine Can Actually Install Windows 11

Before running the installer, Priya reboots into the BIOS/UEFI screen and explicitly enables both TPM (listed as "Intel PTT" on this particular motherboard) and Secure Boot — Chapter 2's own warning that these often ship disabled by default, even on hardware that otherwise fully supports Windows 11. A clean install from bootable USB follows, avoiding any leftover configuration from testing the hardware.

Step 2 — The Account Decision

Priya chooses a Microsoft account during OOBE — deliberately, for the OneDrive integration (Chapter 3) and cross-device Windows Hello (Chapter 4) it enables, accepting Chapter 2's own trade-off of handing over more data by default in exchange for that convenience.

Step 3 — Locking Down the Account Properly

Following Chapter 4's own least-privilege guidance (the same principle *Database Security 3* teaches for database accounts), Priya creates a second, separate local administrator account and switches her daily-use Microsoft account to standard. Windows Hello is set up immediately after — a PIN first, then facial recognition, both backed by the same TPM enabled in Step 1.

Step 4 — Software, the Right Way for Each Tool

Per Chapter 6's own compare-table, Priya installs her everyday apps (browser, messaging) through the Microsoft Store for their sandboxed, auto-updating benefit, and her development tools (a code editor, Git, a database client) via `winget` — scripted in one batch rather than clicked through individually:

> **A REAL WINGET BATCH, CHOSEN DELIBERATELY PER CHAPTER 6'S OWN GUIDANCE**
>
> `winget install Git.Git Microsoft.VisualStudioCode Notepad++.Notepad++` — three developer tools installed in one command, each tracked afterward with `winget upgrade --all` per Chapter 6's own maintenance tip.

Step 5 — Networking & Sharing

Connecting to home Wi-Fi, Priya confirms the network profile is set to **Private**, not Public — Chapter 7's own most commonly cited real-world mistake — specifically so file sharing with another household PC works without a silent, unexplained failure.

Step 6 — Update Discipline & Storage

Chapter 8's own Storage Sense is enabled with a conservative Recycle Bin/downloads retention window, and Active Hours are set to match Priya's actual working schedule, so a feature update never forces a reboot mid-task.

Step 7 — Security, Confirmed Rather Than Assumed

Priya opens the Windows Security app directly (Chapter 9) to confirm Device Encryption activated automatically from the Microsoft account sign-in in Step 2 — exactly the silent activation Chapter 9 warned to check for rather than assume — and saves the recovery key to her Microsoft account, per *Cryptography Fundamentals 11*'s own key-backup discipline.

Step 8 — One Real Script

Chapter 10's own Execution Policy material comes up directly: a small PowerShell script Priya wrote to back up her project folder to an external drive fails on first run with the exact "running scripts is disabled" error. `Set-ExecutionPolicy RemoteSigned` resolves it — the real, recommended middle ground Chapter 10 named, not `Unrestricted`.

Step 9 — The Small Habits

Clipboard history (Chapter 11) is enabled, with the habit of clearing it after copying anything sensitive already in place. Two virtual desktops are set up — one for client work, one for personal projects — each using Snap Layouts independently, exactly as Chapter 11 described the two features working together rather than competing.

CHAPTER-ATTRIBUTION TABLE

- **Chapters 1, 2** — the interface tour and hardware/installation groundwork (Step 1)
- **Chapter 2** — the account-type trade-off (Step 2)
- **Chapter 3** — OneDrive integration enabled by the account choice (Step 2)
- **Chapter 4** — least-privilege accounts and Windows Hello (Step 3)
- **Chapter 5** — the Settings app used throughout, not revisited as its own step
- **Chapter 6** — Store vs. winget, chosen per-tool (Step 4)
- **Chapter 7** — the network profile check (Step 5)
- **Chapter 8** — Storage Sense and Active Hours (Step 6)
- **Chapter 9** — confirming BitLocker/Device Encryption and the recovery key (Step 7)
- **Chapter 10** — Execution Policy resolved in practice (Step 8)
- **Chapter 11** — clipboard history discipline and virtual desktops (Step 9)

HONEST SCOPE NOTE

This setup deliberately stops at the individual-workstation level — no domain join, no Group Policy, no centrally managed update rings, and no deep Registry work. Those are exactly the topics *Windows 11 Troubleshooting & Administration* picks up next: the Registry, Group Policy, Task Scheduler, and real incident diagnosis, building on everything just configured here rather than repeating it.

Hands-On Exercises

EXERCISE 1

Explain why Priya's separate administrator account, created in Step 3, is a genuine application of the same principle Chapter 4 compared to Database Security 3's own least-privilege guidance, rather than an unrelated extra precaution.

View solution

EXERCISE 2

Explain why Priya specifically checked the network profile in Step 5 before troubleshooting anything else about her file-sharing setup, using Chapter 7's own reasoning.

View solution

EXERCISE 3

Using this chapter's own honest scope note, explain what kind of problem on Priya's machine would require Windows 11 Troubleshooting & Administration rather than anything covered in this capstone.

View solution

CHAPTER 12 QUICK REFERENCE

- A real workstation setup touches nearly every chapter of this course, in the order a genuine setup actually hits them — not as isolated topics
- Security decisions (TPM/Secure Boot, account type, least privilege, BitLocker) compound — each step depends on the one before it
- Confirming a setting (Device Encryption's silent activation) is as important as configuring one deliberately
- This course covers the individual workstation; Windows 11 Troubleshooting & Administration covers the deeper administrative layer — Registry, Group Policy, Task Scheduler, and real incident diagnosis