Introduction to Linux

A Beginner's Complete Guide

CONTENTS

Chapter 1 — What is Linux?

If you've ever used a smartphone, streamed a film, searched the web, or paid with a card at a till, you've already relied on Linux — whether you knew it or not. Linux powers more of the world's computers than any other operating system, yet most people have never intentionally installed it. This chapter explains what Linux actually is, where it came from, and why millions of people choose it for everything from writing documents to running the internet.

> **Don't worry about the jargon.** This chapter introduces a few technical terms — kernel, distro, open source — but each one is explained as we go. By the end you'll have a solid mental model of how Linux fits together, which makes everything in the later chapters much easier to follow.

1. A Brief History

To understand Linux you need to know a little about what came before it.

`1969` **Unix is born** at Bell Labs (AT&T). It was a powerful, multi-user operating system written by researchers for researchers. Influential but expensive — universities could get it cheaply, but ordinary people couldn't.

`1983` **Richard Stallman launches the GNU Project** — a plan to build a completely free (as in freedom) Unix-like operating system from scratch. GNU created many essential tools (the compiler, text editor, shell) but never finished its own kernel — the core component that talks to the hardware.

`1991` **Linus Torvalds, a 21-year-old Finnish student**, posts a message to an internet newsgroup: *"I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu)"*. He was writing a kernel for his new PC. That kernel became Linux.

`1992` **GNU + Linux kernel combine** to create a complete, free operating system. The GNU tools gave Linux its software; the Linux kernel gave GNU its missing piece.

`1990s` **Distributions appear** — Slackware, Debian, Red Hat. Groups of developers package the kernel, GNU tools, and additional software into something ordinary people can actually install.

`2004` **Ubuntu launches**, built on Debian, with a mission to make Linux accessible to everyone. It becomes the most popular desktop Linux distro and the one most beginners start with today.

Today

Linux runs everywhere — 96% of the world's top web servers, all 500 of the world's fastest supercomputers, Android phones, smart TVs, in-car systems, and the computers on the International Space Station.

> **The name "Linux"** is a blend of Linus (Torvalds) and Unix. Linus originally wanted to call it "Freax" but the administrator of the server where he uploaded it renamed the folder "Linux" — and the name stuck. The mascot, Tux the penguin, was chosen because Linus was once bitten by a penguin at a zoo and thought they were "cuddly in a geeky way."

2. The Kernel vs the Operating System

People often say "Linux" when they mean "a Linux-based operating system." Technically, Linux is just the **kernel** — and the distinction matters.

What is the kernel?

The kernel is the core program that sits between your software and your hardware. When a program wants to read a file, play a sound, or send data over the network, it asks the kernel — which translates that request into instructions the hardware understands. The kernel is always running, always in control, and users never interact with it directly.

What is the full operating system?

An operating system is the kernel *plus* everything built on top of it: a file manager, a desktop, a web browser, a text editor, system tools, and so on. When you install Ubuntu or Fedora, you're installing the Linux kernel bundled with hundreds of other programs that together form a usable system.

> **GNU/Linux:** Purists sometimes insist on calling it "GNU/Linux" to credit the GNU Project's tools that fill out the OS. Both terms refer to the same thing — most people just say "Linux" and everyone knows what they mean.

3. Open Source and Free Software

Linux is **open source**. This means the source code — the human-readable instructions that make up the program — is publicly available for anyone to read, study, modify, and distribute. This is radically different from Windows or macOS, where the source code is a closely guarded secret.

What "free" actually means

In Linux circles, "free" has two distinct meanings, often summarised as **"free as in speech"** and **"free as in beer."**

- **Free as in speech (libre)** — you have the freedom to use, study, modify, and share the software. This is the important one.
- **Free as in beer (gratis)** — it costs nothing. Most Linux distros are also free to download, though some enterprise versions (like RHEL) charge for support.

Why does open source matter to you?

- **Security** — anyone can audit the code and find vulnerabilities. Bugs get fixed faster because thousands of developers can see and fix them.
- **No lock-in** — if the company behind your software disappears or changes direction, the community can carry it on.
- **Transparency** — you know exactly what the software does. No hidden telemetry, no secret data collection baked in by design.
- **Cost** — no licence fees, no per-seat charges, no subscription just to keep using your own computer.

> **You don't need to read code to benefit from open source.** The fact that *someone* can check the code is what matters. Most Windows and macOS users never read their source code either — but with proprietary software, *nobody outside the company* is allowed to.

4. What is a Distribution (Distro)?

Nobody installs "the Linux kernel" on its own — it's just an engine with no body. A **distribution** (or **distro**) is a complete package: the kernel, a desktop environment, a set of

pre-installed applications, a package manager for installing more software, and an installer to set it all up on your computer.

Think of it like this: the Linux kernel is the engine. A distro is the whole car — bodywork, seats, dashboard, and all — built by different manufacturers to different designs, all using the same engine underneath.

Why are there so many distros?

Because anyone can take the kernel and build a distro from it. Some are built by companies (Ubuntu by Canonical, Fedora by Red Hat), some by communities (Debian, Arch Linux), and some by individuals. Different distros make different choices about:

- **Which desktop to include** — GNOME, KDE, XFCE, and others
- **How software is installed** — apt, dnf, pacman, and other package managers
- **How often it updates** — stable (releases every 2 years) vs rolling (constantly updated)
- **Who it's aimed at** — beginners, power users, servers, specific hardware

> **How many distros are there?** Hundreds — possibly over a thousand if you count every variant. The good news: the core skills you learn on one distro transfer to most others. The terminal commands, the file system layout, and the general concepts are the same everywhere. Chapter 2 will help you pick the right one to start with.

The main distro families

Most distros descend from a handful of "parent" distros. Knowing the family helps because distros in the same family share the same package manager and many conventions:

- **Debian family** — Debian → Ubuntu → Linux Mint, Pop!_OS, Zorin OS, Raspberry Pi OS. Uses the `apt` package manager. The largest family; great for beginners.
- **Red Hat family** — Red Hat → Fedora → CentOS/AlmaLinux/Rocky Linux. Uses `dnf` (formerly `yum`). Dominant in enterprise and server environments.
- **Arch family** — Arch Linux → Manjaro, EndeavourOS. Uses `pacman`. Rolling release; cutting-edge but requires more hands-on management.

- **SUSE family** — openSUSE Leap (stable) and Tumbleweed (rolling). Uses `zypper`. Strong in Europe and enterprise settings.
- **Independent** — Slackware, Gentoo, NixOS. Their own package systems; generally for advanced users.

5. Linux vs Windows vs macOS

LINUX

- Free to download and use
- Open source — fully transparent
- Hundreds of distros to choose from
- Highly customisable
- Excellent for programming and servers
- Longer learning curve initially
- Some hardware/software compatibility gaps
- Strong community support

WINDOWS

- Licence cost built into most PCs
- Closed source — proprietary
- One version (Home/Pro/Enterprise)
- Limited customisation
- Broadest hardware/software compatibility
- Familiar to most users
- More targeted by malware
- Increasing advertising and telemetry

MACOS

- Tied to Apple hardware (expensive)
- Closed source — Unix-based underneath
- Single, polished version
- Limited customisation
- Strong creative/media software ecosystem
- Very stable and well-integrated
- Less targeted by malware than Windows
- Apple controls the full stack

> **You don't have to choose one forever.** It's perfectly normal to run Linux alongside Windows (dual-boot) or inside a virtual machine. Chapter 3 covers both approaches. Many people start by trying Linux in a virtual machine with no risk to their existing setup.

6. Why Do People Choose Linux?

People come to Linux for many different reasons. Here are the most common:

An old PC with a new life

Windows 11 dropped support for a huge range of older hardware. Linux runs well on machines that Windows would leave behind — a ten-year-old laptop with 4 GB of RAM can

run a lightweight Linux distro like Linux Mint XFCE perfectly well, breathing years more useful life into hardware that would otherwise go to landfill.

Privacy and control

Windows 11 ships with advertising IDs, Cortana telemetry, OneDrive integration, and various data-collection features that are difficult or impossible to fully disable. Linux collects nothing by default. Your computer does what you tell it, not what a corporation decided is good for their business model.

Software development

Most professional development tools — web servers, databases, containers, cloud infrastructure — run on Linux. Developing on Linux means your local environment matches production. Most online tutorials, Docker images, and cloud VMs assume Linux. Many developers find Linux simply removes friction.

Security

Linux has a strong security model built in. User permissions, process isolation, and the package manager system (installing software from verified repositories rather than random websites) dramatically reduce the risk of malware. Linux desktops very rarely need antivirus software.

Cost

A full Linux installation — OS, office suite, image editor, video editor, and hundreds of other applications — costs nothing. For schools, charities, small businesses, and individuals, this matters.

Curiosity and learning

Linux teaches you how computers actually work. The terminal, the file system, the process model — understanding these things makes you a better user of *any* operating system, and they're all there to explore on Linux.

7. Linux is Already in Your Life

You may be surprised to discover how many things you use every day run Linux:

ANDROID

Android is built on a modified Linux kernel. About 3 billion active devices.

WEB SERVERS

Google, Facebook, Amazon, Wikipedia — virtually all major websites run on Linux.

SUPERCOMPUTERS

All 500 of the world's fastest supercomputers run Linux.

SMART TVS

Most smart TVs — Samsung, LG, Sony — run Linux under the hood.

CARS

Tesla, BMW, Toyota infotainment and control systems use Linux.

SPACE

The ISS, SpaceX Falcon 9, and NASA's Mars helicopters all run Linux.

GAMING

The Steam Deck runs SteamOS, a Linux-based gaming OS.

BANKING

ATMs, stock exchanges, and banking systems overwhelmingly run Linux.

THE CLOUD

AWS, Google Cloud, and Azure all run primarily on Linux servers.

8. What Does Linux Look Like?

One common misconception is that Linux is just a black screen full of cryptic commands. Modern desktop Linux looks polished, clean, and familiar. Here's what you'll find:

- **A full graphical desktop** — taskbar, application launcher, file manager, notification area, just like Windows or macOS.
- **A web browser** — Firefox comes pre-installed on most distros; Chrome and Edge are available to install.
- **An office suite** — LibreOffice handles Word documents, Excel spreadsheets, and PowerPoint presentations with good compatibility.
- **A terminal** — a text interface for running commands, available when you need it. Chapter 11 covers just enough to be capable.

- **An app store** — most distros have a graphical software centre where you can browse and install applications with a click.

> **You can try Linux without installing anything.** Every major distro can be run as a "live" session directly from a USB stick — your computer boots into a fully functional Linux desktop, you try it out, and when you shut down nothing has changed on your hard drive. This is how most people take their first look, and we cover it in Chapter 5.

Chapter Summary

Concept	What it means
Linux	Strictly, the kernel (the core of the OS) created by Linus Torvalds in 1991. Commonly used to mean any complete Linux-based operating system.
Kernel	The program that connects software to hardware. Always running in the background; users don't interact with it directly.
Open source	The source code is publicly available — anyone can read, modify, and distribute it. Enables transparency, security auditing, and community development.
Distribution (distro)	A complete, installable package of Linux: kernel + desktop + applications + package manager. Examples: Ubuntu, Fedora, Debian, Linux Mint.
Distro families	Debian (apt), Red Hat/Fedora (dnf), Arch (pacman), SUSE (zypper). Distros in the same family share a package manager and many conventions.
Free software	"Free as in speech" (freedom to use/modify/share) and usually "free as in beer" (no cost). Most desktop distros cost nothing to download and use.

Next: Chapter 2 — Choosing your distro. We look at the most popular options and match them to different use cases: best for beginners, development, servers, creative work, gaming, privacy, and older hardware.

Chapter 2 — Choosing Your Distro

One of the first things newcomers notice about Linux is that there isn't just one version to install — there are hundreds. That's one of Linux's great strengths, but it can also be paralysing when you're just trying to get started. This chapter cuts through the noise. You don't need to evaluate every option; you just need to find one that fits your situation.

> **No distro is for life.** Choosing a distro is not a permanent commitment. It's very common to start with one (usually Ubuntu or Mint), get comfortable, and then try others as your confidence grows. The skills you learn on any distro transfer to all the others — the terminal commands, the file system layout, and the core concepts are universal.

1. Why Distro Families Matter

As we saw in Chapter 1, most distros belong to a family that shares a package manager and many conventions. When choosing a distro, the family it belongs to is more important than the exact distro name — because it determines how you install software and what tutorials you can follow.

- **Debian / Ubuntu family** — uses `apt` to install software. The largest ecosystem; the most beginner tutorials and Stack Overflow answers relate to this family. Examples: Ubuntu, Linux Mint, Pop!_OS, Zorin OS.
- **Red Hat / Fedora family** — uses `dnf` (older systems use `yum`). Dominant in enterprise and server environments. Examples: Fedora, AlmaLinux, Rocky Linux.
- **Arch family** — uses `pacman`. Rolling release; always cutting-edge. Examples: Arch Linux, Manjaro, EndeavourOS.
- **SUSE family** — uses `zypper`. Strong in European enterprises. Examples: openSUSE Leap, openSUSE Tumbleweed.

> **Beginner tip:** stick with the Debian/Ubuntu family to start. The sheer number of guides, tutorials, and forum answers written for `apt`-based systems will save you a lot of frustration when you hit a

problem.

2. Best Distro for Each Use Case

Rather than listing every distro, here's a practical guide: what's the best starting point depending on what you actually want to do?

COMPLETE BEGINNERS

Top pick: Linux Mint (Cinnamon)

Also good: Zorin OS, Ubuntu

Linux Mint looks and feels closest to Windows — familiar taskbar, Start menu equivalent, system tray. Cinnamon is the most polished and stable desktop for new users. Based on Ubuntu, so all Ubuntu guides work. Ships with multimedia codecs pre-installed (plays MP3s and videos out of the box without hunting for extras).

SOFTWARE DEVELOPMENT

Top pick: Ubuntu LTS

Also good: Fedora, Pop!_OS, Arch/Manjaro

Ubuntu LTS (Long Term Support — released every 2 years, supported for 5) is the de-facto standard for development. Docker images, cloud VMs, CI pipelines, and most developer documentation target Ubuntu. Fedora is a solid alternative that ships newer software sooner. Pop!_OS adds excellent tiling window management popular with developers.

WEB SERVERS & HOME LABS

Top pick: Ubuntu Server LTS

Also good: Debian, AlmaLinux / Rocky Linux

Ubuntu Server is the most popular choice for VPS hosting and home servers — vast documentation, Canonical security updates for 5 years. Debian is ultra-stable with no corporate backing (suits privacy-conscious self-hosters). AlmaLinux and Rocky Linux are free alternatives to Red Hat Enterprise Linux, ideal if you need RHEL compatibility for professional or enterprise work.

GRAPHICS & CREATIVE WORK

Top pick: Ubuntu Studio

Also good: Fedora Design Suite, KDE Neon

Ubuntu Studio is pre-loaded with creative tools: GIMP (image editing), Inkscape (vector graphics), Blender (3D modelling), Kdenlive and DaVinci Resolve (video editing), Audacity (audio). It uses a low-latency kernel tuned for audio production. Note: Adobe Creative Cloud does not run on Linux natively — if you depend on Photoshop or Illustrator, consider running Linux in a dual-boot alongside Windows.

GAMING

Top pick: Pop!_OS

PRIVACY & SECURITY

Top pick: Fedora

Also good: Debian, Tails (advanced)

Also good: Nobara, Bazzite, SteamOS (Steam Deck only)

Gaming on Linux has improved dramatically thanks to Steam's Proton compatibility layer, which lets many Windows games run on Linux without modification. Pop!_OS provides excellent NVIDIA driver support (a dedicated ISO ships with drivers pre-installed). Nobara and Bazzite are Fedora-based and tuned specifically for gaming performance. Valve's SteamOS runs on the Steam Deck and is based on Arch.

Fedora ships with SELinux enforcing by default, up-to-date security patches, and no unnecessary third-party repositories. Debian collects no telemetry and has a strong community commitment to free software. Tails is a specialist OS designed to run entirely from a USB stick, leave no trace on the computer, and route all traffic through the Tor network — for journalists, activists, or anyone in a genuinely high-risk situation.

♻ OLD OR LOW-SPEC HARDWARE

Top pick: Linux Mint XFCE

Also good: Lubuntu, antiX, Puppy Linux

XFCE is a lightweight desktop that runs comfortably with as little as 512 MB of RAM, though 1–2 GB makes for a much better experience. Lubuntu uses LXQt, which is even lighter. antiX and Puppy Linux are designed for very old hardware (pre-2010 machines, 256–512 MB RAM) and can even run entirely from RAM for a snappy experience on ancient laptops.

🧪 POWER USERS & TINKERERS

Top pick: Arch Linux

Also good: Manjaro, EndeavourOS, Gentoo

Arch Linux gives you a minimal base and lets you build up exactly the system you want. Nothing is installed you didn't choose; nothing runs you didn't configure. The Arch Wiki is legendary — arguably the best Linux documentation on the internet and useful even when you're not running Arch. Manjaro and EndeavourOS make Arch more approachable with a guided installer while keeping the rolling-release, cutting-edge software model.

3. At-a-Glance Comparison

Distro	Family	Difficulty	Release model	Best for
Linux Mint	Debian/Ubuntu	Easy	Point releases (~every 6 months)	Beginners, Windows switchers, older hardware (XFCE edition)
Ubuntu LTS	Debian/Ubuntu	Easy	LTS every 2 years (5 yr support)	Development, general use, servers
Zorin OS	Debian/Ubuntu	Easy	Point releases	Windows/macOS switchers; polished UI

Distro	Family	Difficulty	Release model	Best for
Pop!_OS	Debian/Ubuntu	Easy	Point releases	Developers, gaming (NVIDIA), tiling window management
Ubuntu Studio	Debian/Ubuntu	Easy	LTS + interim releases	Creative / multimedia production
Debian	Debian	Medium	Stable point releases (~every 2 yrs)	Servers, privacy, stability above all else
Fedora	Red Hat	Medium	Point releases (~every 6 months)	Development, bleeding-edge desktop, security
AlmaLinux / Rocky	Red Hat	Medium	Follows RHEL releases	Enterprise servers, RHEL compatibility
Manjaro	Arch	Medium	Rolling (held back for testing)	Up-to-date software, Arch without the manual install
Arch Linux	Arch	Advanced	Rolling (bleeding edge)	Power users who want full control
Lubuntu	Debian/Ubuntu	Easy	Point releases	Very old/low-RAM hardware (< 1 GB RAM)
Tails	Debian	Medium	Point releases	Maximum anonymity; runs from USB, leaves no trace

4. A Closer Look at the Top Beginner Picks

Linux Mint

If you're coming from Windows and want the most comfortable landing zone, Linux Mint is the answer. The Cinnamon desktop will feel immediately recognisable: there's a taskbar at the bottom, a Start-menu-style launcher, a system clock, a notification area. The Update Manager is graphical and polite — it asks before doing anything. The Software Manager lets you browse and install thousands of applications by clicking, no terminal required.

Mint comes in three desktop flavours:

- **Cinnamon** — the flagship; most polished; needs 2 GB RAM minimum (4 GB recommended).
- **MATE** — lighter than Cinnamon; traditional two-panel desktop; good for 1–2 GB RAM.
- **XFCE** — lightest of the three; excellent for older hardware; comfortable at 512 MB RAM.

Ubuntu

Ubuntu is the most widely used Linux desktop and the reference distro for most developer tooling. It uses the GNOME desktop — a modern, clean interface that's different from Windows but intuitive once you spend a few hours with it. Ubuntu has excellent hardware detection and usually "just works" after installation.

Ubuntu releases on a 6-month cycle. The **LTS** (Long Term Support) versions — released in April of even-numbered years (22.04, 24.04…) — receive security patches for five years and are the ones you should install unless you specifically need the very latest software.

Fedora Workstation

Fedora ships whatever is newest — the latest Linux kernel, the latest GNOME, the latest development tools. It's the distro where new Linux features appear first before trickling down to Ubuntu and Mint. Fedora is backed by Red Hat, uses SELinux for enhanced security, and is updated every six months with an 18-month support window. It's an excellent choice for developers who want to stay current.

> **A word on Ubuntu and Snap:** Ubuntu pushes its Snap package system quite aggressively — some applications (including Firefox) are installed as Snaps by default. Snaps start slower and take more disk space than traditional packages. This is a common frustration for Ubuntu users. Linux Mint blocks Snaps by default and uses traditional packages instead, which is one reason many people prefer it.

5. Release Models: Point vs Rolling

Distros update their software in one of two ways, and this affects your experience significantly.

Point releases (stable)

The distro ships a complete, tested version (e.g., "Ubuntu 24.04"). Applications are frozen at a specific version for the lifetime of that release. Security patches arrive but the core software doesn't jump to a new major version. Your system stays predictable and stable; nothing breaks unexpectedly after an update.

- **Pros:** stable, predictable, well-tested before release
- **Cons:** software can be months or years behind the latest version
- **Examples:** Ubuntu LTS, Linux Mint, Debian, Fedora

Rolling releases

There is no fixed version. The distro updates continuously — install it once and keep updating forever. You always have the latest software, but updates occasionally break things.

- **Pros:** always up to date, no major version upgrades to manage
- **Cons:** occasional breakage from untested updates; less suitable for critical systems
- **Examples:** Arch Linux, Manjaro, openSUSE Tumbleweed, Fedora Rawhide

> **For beginners:** stick with a point-release distro. The stability is well worth the slightly older software versions. You can always install a newer version of a specific application via Flatpak if you need it.

6. Hardware Requirements at a Glance

Most modern Linux desktops are less demanding than Windows 11, but requirements vary by desktop environment:

Desktop	Min RAM	Comfortable RAM	CPU	Disk
GNOME (Ubuntu, Fedora)	2 GB	4 GB+	Any 64-bit	25 GB+
KDE Plasma	2 GB	4 GB+	Any 64-bit	20 GB+
Cinnamon (Mint)	1 GB	4 GB+	Any 64-bit	20 GB+
XFCE	512 MB	1–2 GB	Any 64-bit	10 GB+
LXQt (Lubuntu)	256 MB	512 MB–1 GB	Any 64-bit	8 GB+
Server (no desktop)	512 MB	1–2 GB	Any 64-bit	10 GB+

> **32-bit hardware:** almost all modern distros have dropped 32-bit support. If you have a very old machine (pre-2009) with a 32-bit processor, look at antiX, Puppy Linux, or Tiny Core Linux — all still support 32-bit. Check your processor specs if unsure; most processors made since 2007 are 64-bit.

7. How to Try Before You Install

You don't need to install anything to try Linux. There are two common ways to test-drive a distro risk-free:

Live USB (recommended for most people)

Write a Linux ISO to a USB stick, boot from it, and run Linux directly from the USB drive. Everything works — you can browse the web, try the desktop, check that your WiFi and hardware work — without touching your existing Windows installation. When you shut down, your computer is exactly as it was before. Chapter 4 covers creating a live USB in detail.

> **The live session is the real deal.** What you see in the live session is exactly what you'll get after installing. If your WiFi works in the live session, it'll work after installing. If it doesn't, that's a useful thing to know before you commit.

Virtual machine

Install VirtualBox (free) or VMware on your existing Windows or macOS machine, then install Linux inside it. Linux runs in a window alongside your normal desktop. This is excellent for experimenting — you can pause it, take snapshots, and roll back if anything goes wrong. Performance is somewhat reduced compared to running natively, so it's not ideal for gaming or video editing, but it's perfect for learning the basics.

> **VirtualBox** is free and works on Windows, macOS, and Linux. Download it from virtualbox.org, then download a Linux ISO (for example, from ubuntu.com or linuxmint.com), create a new VM, and install Linux as if it were a real machine. The whole process takes about 20 minutes.

8. Making Your Decision

If you're still unsure, answer these three questions:

1. **Is this an old PC with limited RAM?** → Linux Mint XFCE or Lubuntu.
2. **Do you want to learn development or run servers?** → Ubuntu LTS or Fedora.
3. **Everything else (general use, Windows replacement)?** → Linux Mint Cinnamon. It's the most forgiving starting point and the recommendation you'll find from most experienced Linux users when asked "what should a new user install?"

> **The "distro-hopping" trap:** many new Linux users spend weeks comparing distros, reading Reddit threads, and installing and reinstalling different options. This is a known rabbit hole. Pick one of the recommended distros above, install it, and actually use it for a few months — you'll learn far more that way than by reading comparison articles. There's always time to switch later.

Chapter Summary

Use case	Recommended distro	Family / package manager
Complete beginner	Linux Mint (Cinnamon)	Debian/Ubuntu — apt
Software development	Ubuntu LTS or Fedora	Debian/Ubuntu — apt · Red Hat — dnf

Use case	Recommended distro	Family / package manager
Web servers / home lab	Ubuntu Server LTS or Debian	Debian/Ubuntu — apt
Creative / multimedia	Ubuntu Studio	Debian/Ubuntu — apt
Gaming	Pop!_OS or Nobara	Debian/Ubuntu — apt · Red Hat — dnf
Privacy / security	Fedora or Debian	Red Hat — dnf · Debian — apt
Old / low-spec hardware	Linux Mint XFCE or Lubuntu	Debian/Ubuntu — apt
Power users	Arch Linux or Manjaro	Arch — pacman
Try risk-free	Any distro via live USB or VirtualBox	—

Next: Chapter 3 — Preparing to install. Before you put Linux on a real machine you need to check your hardware, understand BIOS vs UEFI, decide between dual-boot and a full install, and back up your data. We'll walk through all of it.

Chapter 3 — Preparing to Install

Installing Linux isn't difficult, but a little preparation beforehand saves a lot of headaches. This chapter covers everything you should do *before* you boot from a USB stick: checking your hardware, understanding how your computer starts up, deciding on a dual-boot or full install, and — most importantly — backing up your data.

> **Back up first, always.** Installing an operating system involves writing to your hard drive. Mistakes happen, and drives can fail. Before you do anything else in this chapter, make sure you have a current backup of any files you can't afford to lose. We cover exactly how to do this in section 2.

1. Checking Your Hardware

Most hardware made in the last ten years works well with Linux. A few categories are worth checking before you commit to an installation.

The live USB test

The single most useful hardware test is to boot from a live USB (covered in Chapter 4) *before* installing. If everything works in the live session — WiFi connects, sound plays, screen resolution is correct, touchpad responds — it will work after installing too. If something is broken in the live session, you know to investigate before installing.

WiFi — the most common sticking point

Most WiFi chips work out of the box. The common exceptions are some Broadcom chips (found in many older laptops) and certain Realtek adapters. If your WiFi doesn't appear in the live session:

- Connect via Ethernet cable temporarily — the installer can then download the proprietary driver.

- Ubuntu and Mint include Broadcom drivers in their "additional drivers" tool, accessible after install.
- Search for your chip model (e.g., "BCM43142 Linux") to find the specific driver package needed.

Graphics cards

- **Intel integrated graphics** — always works perfectly; open-source drivers are built into the kernel.
- **AMD graphics** — works very well; open-source AMDGPU driver is built in and performs excellently.
- **NVIDIA graphics** — the open-source Nouveau driver is installed by default but is limited in performance. For gaming or CUDA work, install NVIDIA's proprietary driver after installation (Ubuntu and Mint make this straightforward via the "Additional Drivers" tool).

Printers and scanners

Most modern printers from HP, Canon, Brother, and Epson work automatically or with a free driver download from the manufacturer's website. Older or cheaper printers using Windows-only drivers (GDI/host-based) will not work. Check the OpenPrinting database (openprinting.org) for your specific model.

Checking your current hardware (on Windows)

Before installing, note down your hardware details from Windows Device Manager (`Win + X → Device Manager`). Specifically:

- WiFi adapter name and chip model
- Graphics card(s)
- Available disk space (right-click drive in File Explorer → Properties)
- Total RAM (System → About)

2. Backing Up Your Data

Do this before anything else. Even if you're only planning a dual-boot alongside Windows, you are resizing partitions — and that carries a small but real risk of data loss if power fails or something goes wrong.

What to back up

- **Documents, photos, music, videos** — anything in your user folder
- **Browser bookmarks** — export from Chrome/Firefox to a file
- **Email** — if you use a desktop email client (Outlook, Thunderbird), export your mailbox
- **Game saves** — often stored in AppData on Windows; back up if important
- **Software licence keys** — for any paid Windows software you might need to reinstall

Where to back up

- **External hard drive or USB stick** — fastest and most reliable option
- **Cloud storage** — OneDrive, Google Drive, Dropbox; good if your internet connection is fast enough
- **Second internal drive** — if your machine has two drives and you're only touching the first one

> **Verify your backup.** Don't just copy files — open a few and confirm they're readable. A backup you haven't tested is not a backup you can trust.

3. BIOS vs UEFI — How Your Computer Starts

When you press the power button, your computer runs a small program stored on the motherboard that gets everything initialised before handing control to the operating system. There are two generations of this program, and knowing which one you have affects how you install Linux.

BIOS (Legacy)

BIOS (Basic Input/Output System) is the older system, present on most computers made before 2012. It's a simple text-based interface. BIOS uses a partition scheme called **MBR** (Master Boot Record) which supports drives up to 2 TB and a maximum of four primary partitions.

UEFI

UEFI (Unified Extensible Firmware Interface) is the modern replacement. Most computers sold since 2012 use UEFI. It has a graphical interface (sometimes touchscreen-capable), supports drives larger than 2 TB, and uses a partition scheme called **GPT** (GUID Partition Table) which supports up to 128 partitions and much larger disks.

UEFI systems have a dedicated small partition called the **EFI System Partition (ESP)**, usually around 100–500 MB, formatted as FAT32. This is where bootloaders live. When you install Linux alongside Windows on a UEFI system, Linux adds its bootloader (GRUB) to the same ESP — both OSes then appear in the boot menu.

> **How to check which you have (Windows):** Press `Win + R`, type `msinfo32`, and press Enter. Look for "BIOS Mode" — it will say either "UEFI" or "Legacy". Alternatively, open Disk Management and look for a "EFI System Partition" — if it's there, you have UEFI.

Which matters for Linux?

Most modern Linux installers handle both automatically. The important thing is to boot your installation media in the *same mode* as your existing Windows installation. If Windows is installed in UEFI mode, boot the Linux installer in UEFI mode. Mixing modes causes bootloader problems that are annoying to fix. Chapter 5 explains how to select the correct boot mode from your computer's boot menu.

4. Secure Boot

Secure Boot is a UEFI feature that checks whether the software being loaded during startup has been digitally signed by a trusted authority. It was designed to prevent malware from hijacking the boot process.

Does Linux work with Secure Boot?

The major distros — Ubuntu, Fedora, Linux Mint, Debian — all include Secure Boot support. Their bootloaders are signed with Microsoft's key (which all major PC manufacturers trust), so they will boot with Secure Boot enabled. For most users installing Ubuntu or Mint, Secure Boot does not need to be touched.

Some smaller or older distros, and some proprietary drivers (notably older NVIDIA drivers), are not Secure Boot compatible. In these cases you may need to disable Secure Boot.

How to disable Secure Boot (if needed)

1. Restart your computer and press the key to enter firmware settings during startup — usually **F2, F10, F12, Del, or Esc** depending on manufacturer. The key is often briefly shown on screen during boot ("Press F2 to enter setup").
2. Navigate to the **Security** or **Boot** tab (varies by manufacturer).
3. Find **Secure Boot** and change it from Enabled to Disabled.
4. Save and exit (usually **F10**). Your computer will restart.

> **Don't disable Secure Boot unless you need to.** If your chosen distro supports it (Ubuntu, Mint, Fedora all do), leave Secure Boot enabled. It provides genuine protection against certain types of rootkits and bootkits.

5. Dual-Boot vs Full Install

Before you install, decide whether you want to keep Windows or replace it entirely.

FULL INSTALL (REPLACE WINDOWS)

PROS

- Simpler setup
- Linux gets the whole drive
- No boot menu to navigate
- No space sharing

CONS

- Windows is gone
- Windows-only software unavailable
- Hard to reverse without reinstalling

DUAL-BOOT (KEEP WINDOWS)

PROS

- Keep Windows as a safety net
- Access Windows-only apps
- Easy to go back if needed

CONS

- More complex to set up
- Split disk space
- Must choose OS at each boot
- Windows updates can break GRUB

> **Recommended for most first-timers:** dual-boot. You keep Windows as a fallback while you get comfortable with Linux. Once you find yourself never booting into Windows, you can reclaim the space with a full install later.

A third option: a separate drive

If your computer has a second hard drive or SSD (or you add one), you can install Linux entirely on that drive and leave the Windows drive untouched. This is the safest option for dual-booting — the two operating systems live on completely separate physical drives. When installing Linux, you just select the second drive. The UEFI boot menu then lets you choose which drive to boot from.

6. Understanding Partitions

A **partition** is a section of a hard drive that the operating system treats as a separate storage area. Think of your drive as a plot of land, and partitions as plots divided by fences — each section is used independently. You can have multiple partitions on one physical drive.

What partitions does Linux need?

At a minimum, Linux needs just one partition to install into. In practice, most installations create two or three:

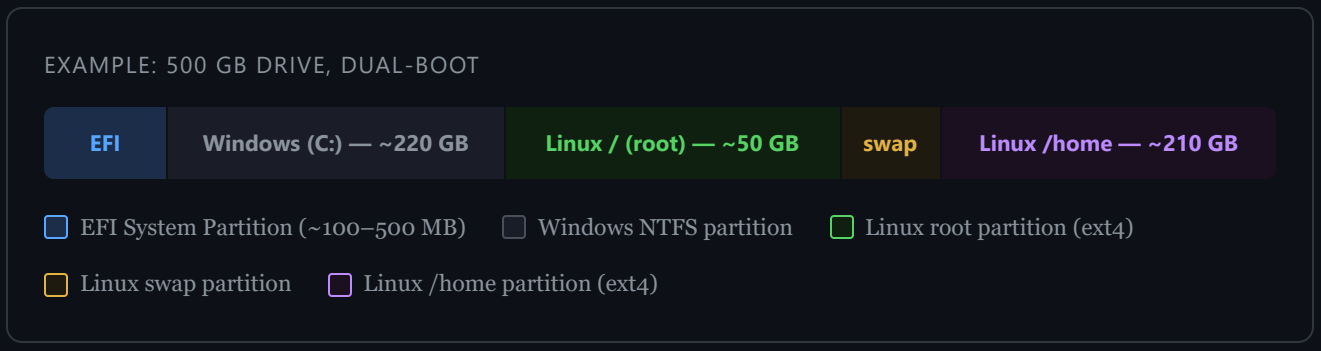
- **/ (root)** — the main Linux partition. Everything lives here: the operating system, programs, and your files. Recommended minimum: 20 GB; 40–60 GB is comfortable.
- **swap** — overflow space used when RAM is full; also used for hibernation. On modern systems with plenty of RAM this is less critical. A common rule: match your RAM size, up to about 8

GB. Most installers create this automatically.

- **/home** — a separate partition for your personal files (documents, music, settings). Keeping /home separate means you can reinstall the OS without losing your files. Optional but recommended on systems with large drives.

On UEFI systems, there's also the EFI System Partition (ESP). If Windows already has one, Linux will use it — you don't need to create a new one.

What a dual-boot disk looks like



File systems

Windows uses **NTFS** for its partitions. Linux uses **ext4** by default — it's fast, reliable, and has been the standard Linux file system for over a decade. Newer options like Btrfs and XFS exist and are used by some distros (Fedora uses Btrfs by default), but ext4 is perfectly fine for a first install. The installer handles all of this automatically.

> **Can Linux read Windows files?** Yes — Linux can read and write to NTFS partitions without any extra setup. If you dual-boot, you can access your Windows files (documents, downloads) from within Linux.

7. Making Space for Linux (Dual-Boot)

If you're dual-booting alongside Windows, you need to shrink the Windows partition to free up unallocated space for Linux. The Linux installer can do this, but it's safer to do it from Windows first so Windows can update its own records cleanly.

1. **Defragment the Windows drive** (HDDs only — skip for SSDs). Open "Defragment and Optimise Drives" from the Start menu and run it on the C: drive. This clusters files together so the partition can be shrunk more effectively.
2. **Disable Fast Startup in Windows.** Go to Control Panel → Power Options → Choose what the power buttons do → Turn off fast startup. Fast Startup leaves the Windows partition in a "hibernated" state that Linux cannot safely write to.
3. **Open Disk Management** in Windows (`Win + X → Disk Management`). Right-click the C: drive and choose **"Shrink Volume"**.
4. **Enter the amount to shrink** in megabytes. For a comfortable Linux install: 50,000 MB (50 GB) is the minimum; 80,000–100,000 MB is better if you have the space. Click Shrink.
5. The freed space will appear as **Unallocated** (black bar) in Disk Management. Leave it unallocated — the Linux installer will create partitions in it.

> **If Windows won't shrink far enough:** some system files (the hibernation file, page file, system restore points) can block shrinking. To recover more space: disable hibernation by running `powercfg /h off` in an Administrator Command Prompt, then try shrinking again.

8. Pre-Install Checklist

Before you move on to creating your installation media, run through this list:

- ☐ **Data backed up** to external drive or cloud
- ☐ **Hardware checked** — WiFi adapter, graphics card, printer compatibility noted
- ☐ **BIOS/UEFI mode noted** (run msinfo32 in Windows if unsure)
- ☐ **Secure Boot status noted** (on or off; leave on if distro supports it)
- ☐ **Chose dual-boot or full install**
- ☐ **If dual-boot:** Fast Startup disabled in Windows; drive shrunk and unallocated space available
- ☐ **Distro chosen** (from Chapter 2)
- ☐ **USB stick ready** — 8 GB or larger, contents backed up (it will be wiped)
- ☐ **Power supply available** — plug in your laptop before installing; don't rely on battery

> **Laptop users:** plug in to mains power before you start. An installation that loses power halfway through can corrupt both your new Linux install and the existing Windows partition. The whole process takes 15–30 minutes — don't risk it on battery.

Chapter Summary

Topic	Key points
Hardware check	Boot the live USB before installing to verify WiFi, sound, and graphics work. NVIDIA may need proprietary drivers after install.
Backups	Back up all personal files before touching the drive. Verify the backup is readable.
BIOS vs UEFI	Check with msinfo32. Boot the installer in the same mode (UEFI or Legacy) as your existing Windows to avoid bootloader problems.
Secure Boot	Ubuntu, Mint, and Fedora support Secure Boot — leave it on. Disable only if your distro or driver requires it.
Install type	Dual-boot keeps Windows alongside Linux (safer for beginners). Full install replaces Windows (simpler setup).
Partitions	Linux needs at minimum a root (/) partition. Swap and /home are optional but recommended. UEFI systems share the existing ESP.
Making space	For dual-boot: disable Fast Startup in Windows, then use Disk Management to shrink C: and leave the freed space as unallocated.

Next: Chapter 4 — Creating installation media. You've chosen your distro and prepared your machine — now it's time to download the ISO, verify it's genuine, and write it to a USB stick using Rufus or balenaEtcher.

Chapter 4 — Creating Installation Media

You've chosen your distro and prepared your machine. Now you need to get Linux onto something you can boot from — a USB stick, a DVD, or in more advanced setups, via the network. This chapter walks through downloading the Linux ISO file, verifying it's genuine, and writing it to installation media. The whole process takes about 15–20 minutes.

> **What you'll need:** a USB stick of at least 8 GB (most Linux ISOs are between 2–5 GB), an internet connection to download the ISO, and either Rufus (Windows) or balenaEtcher (Windows/macOS/Linux). The USB stick will be completely wiped, so save anything on it first.

1. Downloading the ISO

An **ISO file** (also called a disc image) is a single file containing the complete contents of a Linux installation disc — the bootloader, the operating system, the installer, and all the software that comes pre-installed. You download one ISO and write it to a USB stick or DVD.

Where to download

Always download from the official website of the distro you chose. Do not trust third-party download sites — there have been cases where tampered ISOs were distributed containing malware.

- **Linux Mint** — linuxmint.com → Download
- **Ubuntu** — ubuntu.com → Download
- **Fedora** — fedoraproject.org → Download Fedora
- **Debian** — debian.org → Getting Debian → small installation image
- **Pop!_OS** — system76.com/pop → Download (choose NVIDIA or AMD/Intel)
- **Zorin OS** — zorin.com → Download (Core edition is free)

Which edition to pick

Many distros offer several editions. The main choices to understand:

- **Desktop vs Server** — download the Desktop edition unless you're setting up a headless server with no graphical interface.
- **64-bit (AMD64 / x86_64)** — this is what almost all modern computers use, regardless of whether they have an Intel or AMD processor. Download this one.
- **ARM64 (aarch64)** — for ARM-based machines such as Raspberry Pi or Apple Silicon Macs. Only choose this if you know you have ARM hardware.
- **LTS vs interim releases (Ubuntu)** — choose LTS (e.g., 24.04 LTS) for stability.

Using a torrent

Most distros offer their ISO via torrent as well as direct download. Torrent downloads are often faster (especially for popular distros), resume if interrupted, and are automatically verified against the swarm. If you have a torrent client (qBittorrent is free and reputable), use it — it's perfectly legitimate here.

> **Download time estimate:** a typical ISO is 2–4 GB. On a 50 Mbps connection that's about 5–10 minutes. On slower connections, start the download and come back — it doesn't need babysitting.

2. Verifying the ISO

Once downloaded, it's worth verifying that the file wasn't corrupted during the download and hasn't been tampered with. This is done using a **checksum** — a long string of characters generated from the file's contents. If even one byte is different, the checksum changes completely. The distro's website publishes the expected checksum; you generate one from your downloaded file and compare them.

Finding the checksum on the download page

Look on the same page where you downloaded the ISO for a link that says something like "SHA256SUMS", "Verify your download", or "Integrity check". Download or copy that

checksum string — it will look something like:

```
a0d2d56df4e35d78ac2e8d2ab2d6c34a8e65b5...  ubuntu-24.04-desktop-amd64.iso
```

Verifying on Windows

Open PowerShell (search for it in the Start menu) and run:

```
PS> Get-FileHash C:\Users\YourName\Downloads\ubuntu-24.04-desktop-amd64.iso -Algo
```

The `Hash` value in the output should exactly match the checksum published on the download page. If they don't match, the file is corrupted — delete it and download again.

Verifying on Linux (from a live session or existing install)

```
$ sha256sum ubuntu-24.04-desktop-amd64.iso
a0d2d56df4e35d78ac2e8d2ab2d6c34a8e65b5...  ubuntu-24.04-desktop-amd64.iso
```

> **Is verification mandatory?** Strictly speaking, no — most downloads are fine. But if your installation fails with bizarre errors, a corrupted ISO is one of the first things to rule out. It takes 30 seconds and can save a lot of troubleshooting time.

3. Writing the ISO to a USB Stick

You can't just copy the ISO file to a USB stick — the stick needs to be made *bootable*, which means the ISO's contents are written in a specific way that the computer can start from. Several free tools do this for you.

> **The USB stick will be completely erased.** All existing files on the stick will be destroyed when you write the ISO. Save anything you need from it first.

Rufus

balenaEtcher

WINDOWS ONLY — RECOMMENDED

The most popular and capable USB writing tool for Windows. Fast, free, portable (no installation needed — just download and run). Handles almost every Linux distro correctly and offers options for UEFI vs Legacy boot modes.

- Download from: rufus.ie
- Portable version available (no install)
- Supports GPT/MBR partition schemes
- Auto-detects the ISO type
- Windows only

WINDOWS · MACOS · LINUX

Simpler interface than Rufus — three clicks and you're done. Verifies the write automatically after finishing. Excellent choice for macOS users or anyone who wants the most straightforward experience. Slightly slower than Rufus on Windows.

- Download from: etcher.balena.io
- Cross-platform
- Automatic post-write verification
- No partition scheme options (works for most distros)
- Slightly larger download (~150 MB)

4. Step-by-Step: Writing with Rufus (Windows)

1. **Download Rufus** from rufus.ie. Choose the portable version — it's a single .exe file, no installation required. Run it; Windows may ask for administrator permission — click Yes.
2. **Insert your USB stick.** Rufus will detect it automatically and show it in the "Device" dropdown at the top. If you have multiple USB drives connected, double-check you've selected the correct one.
3. **Click "SELECT"** next to the Boot selection field and browse to your downloaded ISO file.
4. **Partition scheme:** Rufus auto-selects based on your system. For modern UEFI computers (most made after 2012) it will choose **GPT**. For older BIOS/Legacy computers, it will choose **MBR**. If unsure, leave the auto-selection.
5. **Leave everything else at the defaults** — file system, cluster size, and volume label don't need to be changed for Linux ISOs.
6. **Click START.** Rufus will warn you that all data on the USB stick will be destroyed — click OK. The progress bar fills as it writes; this typically takes 3–8 minutes.
7. When the status bar shows **"READY"** in green, the USB is done. Close Rufus and safely eject the USB stick. It's ready to boot from.

> **Rufus may ask about writing in "ISO Image mode" vs "DD Image mode".** For most Linux distros, choose **ISO Image mode** — it creates a USB that works for both installing and running a live session. Only choose DD Image mode if the distro's instructions specifically say to, or if ISO mode doesn't boot correctly.

5. Step-by-Step: Writing with balenaEtcher

1. **Download and install balenaEtcher** from etcher.balena.io. On macOS, drag it to Applications. On Windows, run the installer.
2. **Click "Flash from file"** and select your downloaded ISO.
3. **Click "Select target"** and choose your USB stick. Etcher hides internal drives to reduce the risk of accidentally wiping the wrong disk.
4. **Click "Flash!"** and enter your system password if prompted. Etcher writes the ISO, then automatically verifies the result. Both steps together take 5–12 minutes.
5. When Etcher shows **"Flash Complete!"** the USB is ready. Eject it safely.

> **macOS users:** when you eject the USB after writing, macOS may say "The disk you inserted was not readable by this computer" and offer to Initialise, Ignore, or Eject. Click **Eject** — this is normal. macOS can't read the Linux boot format, but your computer's UEFI firmware can.

6. Burning to DVD (Optional)

USB is faster, more reliable, and reusable — but if your machine has a DVD drive and no USB ports that work at boot time, a DVD is a valid fallback. You'll need a blank DVD-R or DVD+R disc (not a CD — Linux ISOs are too large).

On Windows

Windows 10 and 11 have built-in ISO burning. Right-click the ISO file in File Explorer and select **"Burn disc image"**. Insert a blank disc, select the correct burner drive, and click Burn. Write speed doesn't matter much — slower (4x) produces more reliable results than maximum speed.

On macOS

Open Disk Utility, go to File → Open Disk Image and select your ISO, then File → Burn to disc.

> **DVD vs USB speed:** booting and installing from a DVD takes significantly longer than from a USB stick — often 2–3× longer. USB is always preferable if you have the option.

7. Network Boot / PXE (Advanced)

PXE (Preboot Execution Environment) boot allows a computer to boot from a Linux installer delivered over the network, with no USB stick or DVD involved. The computer requests a boot image from a server on the local network and loads the installer directly into RAM.

This is primarily useful when:

- You're deploying Linux to many machines at once (common in offices and data centres)
- The machine has no working USB ports or optical drive
- You want a centralised install process for a home lab

How PXE boot works (overview)

- The target computer's UEFI/BIOS sends a broadcast DHCP request when set to network boot.
- A DHCP server responds with the address of a TFTP server and the boot file name.
- The computer downloads the bootloader (e.g., PXELINUX or GRUB) over TFTP.
- The bootloader presents a menu; the selected OS image is downloaded over HTTP or NFS.
- The installer runs entirely in RAM.

> **PXE setup requires a server running DHCP, TFTP, and HTTP/NFS services** — typically an existing Linux machine. Tools like **FOG Project** or **Netboot.xyz** package all of this into a manageable solution. PXE is a topic for later in your Linux journey, not something to tackle on a first install.

8. Booting from Your USB Stick

With your USB stick written, the final step in this chapter is to confirm you can boot from it. This is also how you'll start the actual installation in Chapter 5.

The boot menu

Most computers let you choose a temporary boot device by pressing a function key immediately after the power button. The key varies by manufacturer — it's often shown briefly on screen during startup:

Manufacturer	Boot menu key	UEFI/BIOS setup key
Dell	F12	F2
HP	F9 or Esc then F9	F10 or Esc
Lenovo	F12 or Fn+F12	F2 or Fn+F2
ASUS	F8 or Esc	F2 or Del
Acer	F12	F2 or Del
MSI	F11	Del
Gigabyte	F12	Del or F2
Apple Mac	Hold Option (⌥) at startup	—

1. **Insert the USB stick** before powering on.
2. **Power on (or restart)** and immediately press the boot menu key for your manufacturer — usually within the first two seconds of startup. Tap it repeatedly rather than holding it.
3. A boot menu appears listing available devices. **Select your USB stick** — it may appear as the brand name, "USB HDD", "USB Storage Device", or similar. If you see two entries for the same stick (one with "UEFI:" prefix and one without), choose the **UEFI:** version on modern computers.
4. The Linux boot menu appears — usually offering "Try" or "Start" (live session) and "Install". For now, choose **Try** to test hardware compatibility before committing.

> **Nothing happens after selecting the USB?** Common causes: the USB wasn't written correctly (try writing it again), you selected the wrong boot mode (UEFI vs Legacy), or Secure Boot is blocking an unsigned bootloader. Try the other mode entry in the boot menu, or temporarily disable Secure Boot in UEFI settings as described in Chapter 3.

> **If there's no boot menu key:** go into UEFI/BIOS setup (see the table above for keys) and look for a "Boot Order" or "Boot Priority" section. Move USB to the top of the list, save, and restart. Remember to change the boot order back after installing.

Chapter Summary

Task	Key points
Download the ISO	Always download from the official distro website. Choose 64-bit (AMD64). For Ubuntu, choose the LTS version.
Verify the checksum	Compare the SHA256 hash of your downloaded file against the one published on the download page. Use PowerShell on Windows (`Get-FileHash`) or `sha256sum` on Linux.
Write to USB (Windows)	Rufus (rufus.ie) is the best choice. Select your USB, select the ISO, leave defaults, click START. Takes 3–8 minutes.
Write to USB (cross-platform)	balenaEtcher (etcher.balena.io) — three clicks, auto-verifies after writing. Works on Windows, macOS, Linux.
DVD burning	Right-click the ISO in Windows File Explorer → "Burn disc image". Slower than USB; only use if USB booting isn't available.
PXE / network boot	Advanced technique for multi-machine deployments; requires a server with DHCP, TFTP, and HTTP/NFS. Not needed for a first install.
Booting the USB	Press the boot menu key (F12, F9, F8 — varies by manufacturer) immediately after power-on. Select the USB entry with "UEFI:" prefix on modern systems.

Next: Chapter 5 — The installation process. Your USB stick is ready and boots correctly. Now we walk through the actual installation step by step, using Ubuntu as the reference, with notes on where Fedora and Linux Mint differ.

Chapter 5 — The Installation Process

Your USB stick is written, your data is backed up, and you've confirmed the live session boots correctly. Now it's time to install. This chapter walks through the full installation step by step using **Ubuntu 24.04 LTS** as the reference — it's the most widely used distro and its installer is representative of what you'll find on Linux Mint and most Debian-based systems. Differences for Linux Mint, Fedora, and Debian are noted at the end.

> **Time to complete:** the installation itself takes 10–20 minutes on most modern hardware. An SSD installs significantly faster than a traditional spinning hard drive. Plan for about 30 minutes end-to-end including reboots.

1. The Live Session

When you boot from the USB stick, Ubuntu gives you a choice: **Try Ubuntu** or **Install Ubuntu**. Always choose **Try Ubuntu** first.

The live session boots into a fully functional Ubuntu desktop running entirely from the USB stick. Nothing is written to your hard drive at this point. Use this opportunity to:

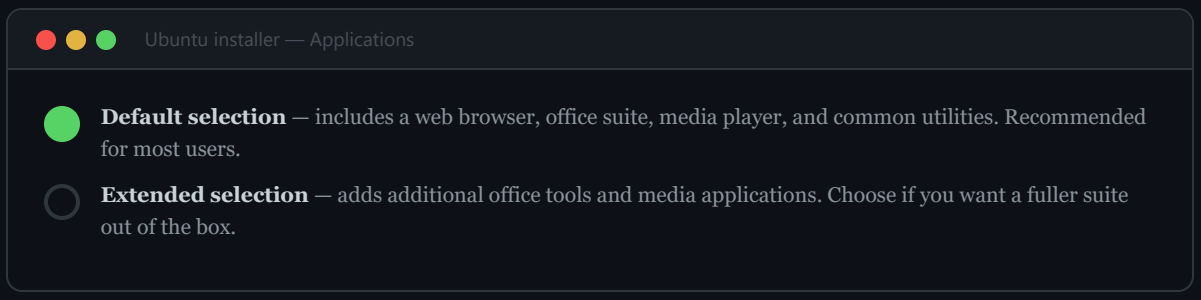
- **Check WiFi** — click the network icon in the top-right corner. If your WiFi networks appear and you can connect, it will work after installing.
- **Check sound** — open a video in Firefox or play audio via the Files app.
- **Check display resolution** — it should match your monitor's native resolution. If it's wrong, it may need attention after installing (covered in Chapter 6).
- **Test the touchpad** (laptops) — scroll, tap, and two-finger scroll.

Once you're satisfied everything works, find the **"Install Ubuntu"** icon on the desktop or in the application menu and double-click it to launch the installer.

> **No WiFi in the live session?** Don't cancel the install — connect via Ethernet cable instead and the installer will download any missing drivers. Proprietary WiFi drivers can be installed after the fact through Ubuntu's "Additional Drivers" tool, which we cover in Chapter 6.

2. The Ubuntu Installer — Step by Step

1. **Language selection.** The installer opens with a language picker. Select your language and click **Next**.
2. **Accessibility.** Ubuntu 24.04 includes an accessibility options screen — screen reader, large text, high contrast. Configure these if needed, or click **Next** to skip.
3. **Keyboard layout.** The installer usually detects your keyboard correctly. Verify by typing a few characters — especially any special keys that differ between layouts (@ symbol, £ sign, accented characters). Click **Next**.
4. **Network connection.** If you're connected to WiFi or Ethernet, the installer will offer to download updates during installation. If you have a fast connection, tick "Download updates while installing Ubuntu" — it saves a step after install. If your connection is slow or metered, skip it and update afterwards.
5. **Type of installation — Normal or Minimal.** Ubuntu 24.04 offers two options:

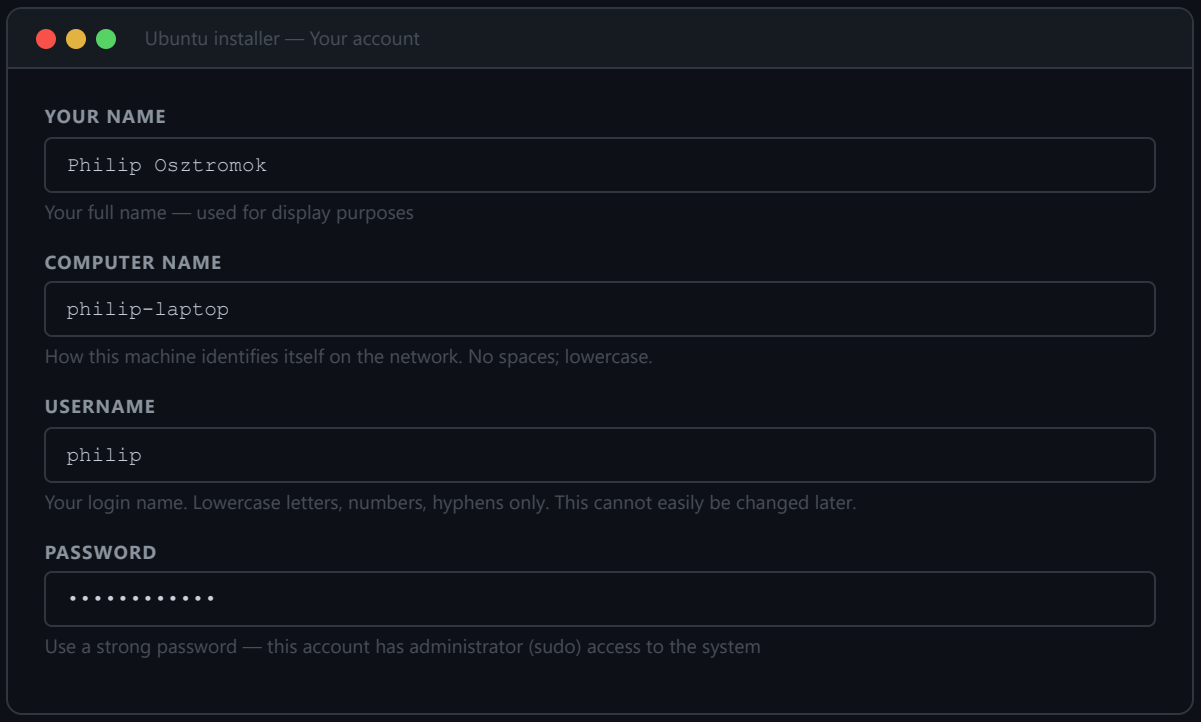


   Choose **Default selection** unless you specifically want the extras. You can always install more software afterwards.

6. **Third-party software and drivers.** You'll see a checkbox: "Install recommended proprietary software". Tick this — it enables installation of NVIDIA drivers, WiFi firmware, and multimedia codecs (MP3 playback, H.264 video) that can't ship as open-source. Without it, some hardware and media won't work immediately after install.
7. **Partitioning — the most important screen.** See Section 3 below for a detailed explanation of the options here. For most people: if dual-booting, choose "Install Ubuntu alongside Windows"; if doing a full install, choose "Erase disk and install Ubuntu".

8. **Time zone.** The installer shows a world map. Click your location or type your city. This sets your system clock and the time zone used for scheduled tasks.

9. **Create your user account.**



Choose your username carefully — it becomes your home directory name (`/home/philip`) and is used throughout the system. It cannot be easily changed after installation.

10. **Review and confirm.** Ubuntu 24.04 shows a summary screen listing all your choices before making any changes to your disk. Read through it carefully — especially the partitioning section. If anything looks wrong, use the Back button. Click **Install** when you're satisfied.

11. **Installation runs.** A progress bar advances as files are copied and configured. This takes 10–20 minutes. The computer does not need your attention during this time. Don't interrupt it or let the laptop battery die.

12. **Restart now.** When installation completes, a dialog asks you to restart. Click **Restart Now**. The installer will tell you to remove the USB stick and press Enter — do so. The computer reboots into your new Linux installation.

3. Partitioning Options Explained

The partitioning screen is where most first-timers feel nervous. Here are the options you'll see and when to choose each one.

Install alongside Windows

Erase disk and install Ubuntu

The installer detects Windows and shrinks it to make space. It creates the Linux partitions automatically. A slider lets you adjust how much space each OS gets. **Choose this for dual-boot if you already made space in Chapter 3, or let the installer do it.**

Wipes the entire selected disk and installs Ubuntu on it. The simplest option — Ubuntu handles all the partition creation for you. **Choose this for a full install replacing Windows. Warning: destroys all existing data on the disk.**

Manual partitioning

Shows a full partition editor where you create, resize, and assign partitions yourself. Needed if the automatic options don't suit your situation — for example, installing to specific partitions on a drive with an unusual layout, or creating a separate /home partition. **Recommended only if you're comfortable with the concepts from Chapter 3.**

LVM / Encryption options

Ubuntu offers LVM (Logical Volume Manager) and full-disk encryption (LUKS). LVM makes it easier to resize partitions later. Encryption protects your data if the laptop is stolen — at the cost of a passphrase on every boot. Both are optional and can be combined. **Encryption is worthwhile on laptops that leave the house.**

Using manual partitioning for dual-boot

If "Install alongside Windows" doesn't appear (sometimes happens with unusual disk layouts), use manual partitioning. The unallocated space you freed in Chapter 3 will appear as free space. Create the following partitions in it:

- **Root partition (/)** — select the free space, click +, set size (minimum 20 GB, ideally 40–50 GB), file system: `ext4`, mount point: `/`
- **Swap partition** — size equal to your RAM (up to 8 GB), file system: swap area
- **/home partition (optional)** — remaining free space, file system: `ext4`, mount point: `/home`
- **EFI partition** — on UEFI systems, select the existing Windows EFI partition and mark it as "boot" / EFI System Partition without formatting it. Do *not* create a new one or format the existing one — that would break Windows boot.

> **In manual mode, never format the Windows partition or the EFI partition.** In the partition editor, the "Format" checkbox must be *unticked* for both the Windows NTFS partition (C:) and the existing EFI System Partition. Only format the new partitions you create in the unallocated space.

4. Full-Disk Encryption (Optional but Recommended for Laptops)

If you select "Erase disk and install Ubuntu" and tick **"Encrypt the new Ubuntu installation for security"**, the entire Linux partition is encrypted using LUKS (Linux Unified Key Setup). A passphrase is set during installation; every time the machine boots, you type this passphrase before Linux loads.

- **Benefit:** if the laptop is stolen or lost, the data on it cannot be read without the passphrase — even if the thief removes the drive.
- **Drawback:** forgotten passphrase = unrecoverable data. There is no bypass and no recovery email. Write the passphrase down and store it somewhere safe and offline.
- **Performance impact:** negligible on modern hardware with AES acceleration.

> **Enable encryption on laptops.** Desktop computers that never leave your home may not need it, but any machine that travels — work laptop, university computer — should be encrypted. The setup cost is one extra passphrase at boot.

5. First Boot

After the installer finishes and the machine restarts, you'll see the **GRUB boot menu** — a text or graphical menu listing all available operating systems. On a dual-boot system this will show Ubuntu and Windows Boot Manager. The default selection (Ubuntu) counts down from 10 seconds and boots automatically if you don't press anything.

Ubuntu loads to a login screen. Enter the password you set during installation.

What you'll see on first login

Ubuntu may launch a welcome screen offering to set up online accounts (Ubuntu One, Google, etc.), configure Ubuntu Pro, and take a quick tour. All of these are optional. You can close the welcome screen immediately if you prefer.

Checking the GRUB menu (dual-boot)

On a dual-boot system, the GRUB menu defaults to Ubuntu. To boot Windows, use the arrow keys to select "Windows Boot Manager" and press Enter. GRUB remembers the last selection by default on some distros — if you find it always boots the wrong OS, you can change the default in `/etc/default/grub` (covered in Chapter 6).

> **Windows may run chkdsk on first boot after install.** This is normal — Windows noticed that something modified its partition table and runs a disk check as a precaution. Let it complete; don't interrupt it.

6. If Something Goes Wrong

Installation problems are rare but they do happen. Here are the most common issues and what to do about them.

"No bootable device found" after installing

The computer can't find a bootloader. Most common cause on UEFI systems: the installer ran in Legacy/BIOS mode but the computer is set to UEFI, or vice versa. Go into UEFI settings and look for a Boot Override or Boot Order section — Ubuntu's GRUB should appear as an entry. If it doesn't, boot from the USB stick again and select "Try Ubuntu", then run `boot-repair` (install it from the terminal with `sudo apt install boot-repair`, then run it).

GRUB menu doesn't appear (dual-boot)

Windows Boot Manager may have taken priority over GRUB in the UEFI boot order. Go into UEFI settings, find the Boot Order, and move "ubuntu" above "Windows Boot Manager". Save and restart.

Installation freezes or crashes

Usually caused by a corrupted ISO or a bad USB write. Verify the ISO checksum (Chapter 4), rewrite the USB stick, and try again. If it crashes at the same point repeatedly, check

whether your RAM is faulty by running the memory test available in the GRUB menu of the live USB.

"Ubuntu" entry missing from GRUB after a Windows update

Windows updates sometimes overwrite the UEFI boot order. Boot from your Ubuntu USB stick in live mode, open a terminal, and run `sudo update-grub` after mounting the installed system. The `boot-repair` tool mentioned above automates this process.

7. Differences for Other Distros

Aspect	Ubuntu 24.04	Linux Mint 21	Fedora 40	Debian 12
Installer name	Ubiquity / new Flutter installer	Ubiquity (same as older Ubuntu)	Anaconda	Debian Installer
Live session first?	Yes — Try before installing	Yes — Try before installing	Yes — Try Fedora option	No — boots straight to installer
Partitioning screen	Simple options + manual mode	Identical to Ubuntu's options	Anaconda's disk configuration tool — different UI but same concepts	Text-based; more detailed but more options
Default file system	ext4	ext4	Btrfs (with subvolumes)	ext4
Codecs & drivers	Optional tick-box during install	Included by default in Mint	Must install RPM Fusion after install for MP3/H.264	Non-free firmware ISO required for some hardware
Encryption option	During install (LUKS)	During install (LUKS)	During install (LUKS)	During install (LUKS)
Difficulty level	Easy	Easy (slightly friendlier)	Medium (Anaconda is less intuitive for beginners)	Medium–Hard (text installer, more decisions)

A note on Fedora's Anaconda installer

Fedora uses its own installer called Anaconda, which looks quite different from Ubuntu's. The main difference is that you configure each section (time zone, partitioning, user account) by clicking tiles on a hub screen, rather than stepping through them sequentially. The concepts are identical — it just takes a moment to get used to the layout. Notably, partitioning in Anaconda requires you to click "Done" twice to confirm — easy to miss.

A note on Debian's installer

Debian's installer is text-based and asks more questions than Ubuntu's, including selecting which software groups to install (desktop, web server, etc.). For a desktop install, choose "Debian desktop environment" plus the specific desktop (GNOME, KDE, etc.). Debian also has a separate "non-free firmware" ISO that includes proprietary WiFi and GPU drivers — download that version rather than the standard ISO for the smoothest hardware experience.

Chapter Summary

Step	What to do
Live session	Choose "Try Ubuntu" first. Verify WiFi, sound, and display before launching the installer.
Partitioning	Dual-boot: "Install alongside Windows". Full install: "Erase disk and install Ubuntu". Manual only if needed.
Third-party software	Tick the box for proprietary drivers and codecs during install. Saves setup time afterwards.
Encryption	Enable LUKS encryption for any laptop that leaves the house. Record the passphrase somewhere safe — it cannot be recovered.
Username	Choose your username carefully — it becomes your home directory path and is difficult to change later.
First boot	GRUB menu appears for dual-boot. Ubuntu loads; close the welcome screen and log in. Windows may run chkdsk once — let it finish.

Step	What to do
Troubleshooting	Boot problems: check UEFI boot order. Missing GRUB: use boot-repair from live USB. Crashes during install: verify the ISO checksum and rewrite the USB.

Next: Chapter 6 — First boot and essential setup. Linux is installed and running. Now we run the first update, fix any hardware issues, set the correct display resolution, configure the hostname, and create additional user accounts.

Chapter 6 — First Boot & Essential Setup

Linux is installed and you're logged in. Before you start using it day-to-day, there are a handful of tasks worth doing immediately — updating the system, checking that all hardware is working, and configuring a few basic settings. None of this is complicated, and most of it takes only a few minutes. Think of this as the "out of the box" setup you do with any new computer.

> **GUI or terminal?** This chapter shows both approaches where available. For a first install, the graphical tools are perfectly fine. Terminal commands are shown alongside them because you'll start seeing them in guides and forum answers — it's useful to recognise what they do even if you don't type them yet. Chapter 11 covers the terminal properly.

1. Run the First Update

The ISO you installed from was built weeks or months ago. Security patches and bug fixes have accumulated since then. The first thing to do on any new Linux install is update everything.

Ubuntu / Linux Mint — graphical method

A notification usually appears shortly after first login offering to install updates. If not, open the **Update Manager** (search for it, or find it in the taskbar/system tray). Click **Install Updates** and enter your password when prompted. This may take 5–15 minutes depending on how much has changed since the ISO was built.

Ubuntu / Linux Mint — terminal method

```
$ sudo apt update && sudo apt upgrade -y
```

- `sudo` — runs the command with administrator privileges (you'll be asked for your password)

- `apt update` — fetches the latest list of available packages from the repositories
- `apt upgrade` — downloads and installs all available updates
- `-y` — automatically answers "yes" to any confirmation prompts

Fedora — terminal method

```
$ sudo dnf upgrade --refresh -y
```

> **Restart after updating.** If the kernel (the core of the OS) was updated, you need to reboot to use the new version. Ubuntu and Mint will show a notification asking you to restart. On Fedora, `dnf` tells you if a reboot is needed.

2. Install Additional Drivers

Even if you ticked the "third-party software" option during installation, Ubuntu and Mint have a dedicated tool for managing proprietary drivers — worth checking after the first update.

Ubuntu / Linux Mint — Additional Drivers tool

1. Open **Software & Updates** (search for it) and click the **Additional Drivers** tab. On Mint it may be listed directly as "Driver Manager" in the application menu.
2. The tool scans your hardware and lists any devices that have proprietary drivers available. Common examples:

   NVIDIA graphics — select the recommended driver (highest version number without "open" or "server" in the name for general desktop use)

   Broadcom WiFi — select the listed driver if your WiFi isn't working

   AMD graphics — usually no action needed; open-source driver is built in
3. Click **Apply Changes** and wait for the driver to install. **Restart when prompted** — driver changes require a reboot to take effect.

Fedora — RPM Fusion and NVIDIA

Fedora doesn't ship proprietary drivers by default. To get NVIDIA drivers, first enable RPM Fusion (a third-party repository):

```
# Enable RPM Fusion free and non-free repos
$ sudo dnf install \
    https://mirrors.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fe
    https://mirrors.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-$(rpm

# Then install the NVIDIA driver
$ sudo dnf install akmod-nvidia
```

> **After installing NVIDIA drivers, reboot and wait.** The driver module is compiled for your specific kernel during the first boot after install. This takes 2–5 minutes; the screen may go blank. Don't force-restart — let it complete. Subsequent boots are normal speed.

3. Locale and Timezone

You set these during installation, but it's worth verifying they're correct — especially if your clock is showing the wrong time.

Graphical — Ubuntu / Mint

Open **Settings → Region & Language** to check your locale (language and date/number formats). Open **Settings → Date & Time** to verify the timezone and confirm that "Automatic Date & Time" (NTP sync) is turned on.

Terminal — check current settings

```
$ timedatectl
               Local time: Mon 2026-06-15 14:32:10 BST
           Universal time: Mon 2026-06-15 13:32:10 UTC
                 RTC time: Mon 2026-06-15 13:32:10
                Time zone: Europe/London (BST, +0100)
System clock synchronized: yes
              NTP service: active
```

Terminal — change timezone

```
$ sudo timedatectl set-timezone Europe/London
# List all available timezones:
$ timedatectl list-timezones | grep Europe
```

Dual-boot clock problem

If you dual-boot with Windows, you may notice that the clock is wrong after switching between the two OSes. This happens because Windows stores the hardware clock in local time, while Linux stores it in UTC. The fix is to tell Linux to also use local time:

```
$ sudo timedatectl set-local-rtc 1 --adjust-system-clock
```

> **Alternatively** (the cleaner fix): configure Windows to store the clock in UTC instead. In Windows, open Registry Editor, navigate to `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\TimeZoneInformation`, and add a DWORD value `RealTimeIsUniversal` set to `1`.

4. Display Resolution and Scaling

Linux should detect your monitor's native resolution automatically. If the display looks blurry, text appears too large, or the resolution doesn't match your monitor's specification, fix it here.

Graphical — Ubuntu (GNOME)

Open **Settings** → **Displays**. You'll see your monitor(s) listed with the current resolution. Click on the resolution dropdown and select the highest available option — this is your monitor's native resolution. For 4K monitors on a laptop, also look at the **Scale** setting — 200% is usually correct for a 4K laptop display.

Graphical — Linux Mint (Cinnamon)

Open **System Settings → Display**. Select your monitor and choose the correct resolution. Mint also has a "Use system font size" option under Display if text appears too small.

Terminal — list available resolutions and change

```
$ xrandr
Screen 0: minimum 320 x 200, current 1920 x 1080, maximum 16384 x 16384
HDMI-1 connected primary 1920x1080+0+0 (normal left inverted right x axis y axis)
   1920x1080     60.00*+  50.00    59.94
   1280x720      60.00    50.00    59.94
   1024x768      60.00

# Set resolution to 1920x1080 at 60Hz:
$ xrandr --output HDMI-1 --mode 1920x1080 --rate 60
```

> **NVIDIA users:** if you just installed the NVIDIA proprietary driver and the resolution is still wrong after rebooting, open **NVIDIA X Server Settings** from the application menu. Go to "X Server Display Configuration" and set the correct resolution there, then click "Save to X Configuration File".

HiDPI / 4K monitors

On GNOME (Ubuntu, Fedora), fractional scaling (e.g., 125%, 150%) can be enabled via:

```
$ gsettings set org.gnome.mutter experimental-features "['scale-monitor-framebuff
```

Then return to Settings → Displays where fractional scale options will now appear.

5. Setting the Hostname

The hostname is the name your computer uses to identify itself on the network and in the terminal prompt. You set it during installation, but you can change it at any time.

Check current hostname

```
$ hostnamectl
 Static hostname: philip-laptop
       Icon name: computer-laptop
         Chassis: laptop 💻
      Machine ID: a1b2c3d4e5f6...
         Boot ID: 1234abcd...
Operating System: Ubuntu 24.04.1 LTS
          Kernel: Linux 6.8.0-39-generic
    Architecture: x86-64
```

Change the hostname

```
$ sudo hostnamectl set-hostname my-new-hostname
```

The change takes effect at the next login (open a new terminal to see it in the prompt).

> **Hostname rules:** use only lowercase letters, numbers, and hyphens. No spaces, no underscores, no dots (those are for fully-qualified domain names). Keep it short and descriptive — `philip-laptop`, `home-server`, `pi-desk`.

6. Creating Additional User Accounts

If other people will use the machine, each person should have their own account. Linux user accounts are completely separate — each has their own home folder, settings, and files.

Graphical — Ubuntu

1. Open **Settings → Users**.
2. Click **Unlock** in the top-right and enter your password.
3. Click **Add User**. Choose the account type:

   Standard — can use the computer normally, can't install system-wide software or change system settings. Suitable for children or guests.

   Administrator — has sudo access; can install software and change system settings. Use for other trusted adults who share the machine.

4. Enter the full name and username, set a password, and click **Add**.

Graphical — Linux Mint

Open **System Settings → Users and Groups**. Click **Add**, enter the details, and set the account type.

Terminal method (all distros)

```
# Create a new user
$ sudo adduser sarah
Adding user 'sarah' ...
Adding new group 'sarah' (1001) ...
Adding new user 'sarah' (1001) with group 'sarah' ...
Creating home directory '/home/sarah' ...
New password:

# If the new user needs administrator (sudo) access:
$ sudo usermod -aG sudo sarah      # Ubuntu/Debian
$ sudo usermod -aG wheel sarah     # Fedora/Red Hat
```

> **Only grant administrator access to accounts that need it.** Every administrator account is a potential attack surface. A standard account can't accidentally (or intentionally) break the system — it simply doesn't have the permissions.

7. Changing the GRUB Default OS (Dual-Boot)

If you dual-boot and find yourself always choosing Windows from the GRUB menu, you can change the default. GRUB's configuration is in `/etc/default/grub`.

```
$ sudo nano /etc/default/grub
```

Look for the line:

```
GRUB_DEFAULT=0
```

`0` means the first entry in the boot menu (usually Ubuntu/Linux). To change it, either replace `0` with the number of the entry you want (counting from 0), or use `saved` to remember the last choice:

```
# Always boot the second entry (index 1 = usually Windows on dual-boot):
GRUB_DEFAULT=1

# Or remember the last OS you chose:
GRUB_DEFAULT=saved
GRUB_SAVEDEFAULT=true
```

After editing, save the file (`Ctrl+O`, then `Ctrl+X` to exit nano) and update GRUB to apply the change:

```
$ sudo update-grub
```

> **GRUB timeout:** you can also adjust how long the boot menu waits before auto-booting. Find the line `GRUB_TIMEOUT=10` and change the number (in seconds). Set it to `0` to skip the menu entirely (not recommended on a dual-boot system), or `-1` to wait indefinitely until you make a choice.

8. First-Boot Checklist

- ☐ **Run system updates** — Software Updater / Update Manager, or `sudo apt update && sudo apt upgrade -y`
- ☐ **Restart after updates** if a new kernel was installed
- ☐ **Check Additional Drivers** — install NVIDIA or Broadcom drivers if listed
- ☐ **Verify display resolution** — Settings → Displays; set to native resolution
- ☐ **Check timezone and clock** — Settings → Date & Time; NTP sync on
- ☐ **Fix dual-boot clock** if time is wrong after switching from Windows — `sudo timedatectl set-local-rtc 1`
- ☐ **Set hostname** — `sudo hostnamectl set-hostname your-name` if you want to change it
- ☐ **Create additional user accounts** if others will use the machine
- ☐ **Set GRUB default** if you prefer Windows to be the default boot option
- ☐ **Install multimedia codecs** if you didn't during installation (see below)

Installing multimedia codecs (if missed during install)

If you didn't tick the third-party software option during installation, add codec support now:

```
# Ubuntu — install restricted extras (MP3, H.264, DVD, fonts)
$ sudo apt install ubuntu-restricted-extras

# Linux Mint — usually pre-installed; if not:
$ sudo apt install mint-meta-codecs

# Fedora — after enabling RPM Fusion:
$ sudo dnf install gstreamer1-plugins-{bad-\*,good-\*,base} gstreamer1-plugin-ope
```

9. Distro-Specific Notes

LINUX MINT DIFFERENCES

- Update Manager has a "trust level" system — Level 1–2 updates are safest; you can safely install all levels
- Multimedia codecs are usually installed by default
- Driver Manager is a separate application in the menu (not inside Software & Updates)
- The Welcome Screen on first boot has useful quick-links to all these settings

FEDORA DIFFERENCES

- Use `sudo dnf upgrade --refresh` instead of `apt upgrade`
- NVIDIA drivers require RPM Fusion — not available without it
- Btrfs filesystem: Fedora creates automatic snapshots with Timeshift or Snapper
- SELinux may block some operations; errors include "Permission denied" even with sudo — check `journalctl` for AVC denials

Chapter Summary

Task	How to do it
System update	**Ubuntu/Mint:** Update Manager GUI, or `sudo apt update && sudo apt upgrade -y`. **Fedora:** `sudo dnf upgrade --refresh -y`. Restart if kernel updated.
Additional drivers	Ubuntu/Mint: Software & Updates → Additional Drivers tab (or Driver Manager on Mint). Fedora: RPM Fusion + `sudo dnf install akmod-nvidia`.

Task	How to do it
Timezone	Settings → Date & Time (GUI), or `sudo timedatectl set-timezone Region/City`. Fix dual-boot clock with `sudo timedatectl set-local-rtc 1`.
Display resolution	Settings → Displays (GNOME) or System Settings → Display (Cinnamon). Use `xrandr` in terminal. Enable fractional scaling via `gsettings` on GNOME.
Hostname	`sudo hostnamectl set-hostname new-name`. Takes effect at next login.
Additional users	Settings → Users (Ubuntu) or System Settings → Users and Groups (Mint). Terminal: `sudo adduser username`. Add sudo: `sudo usermod -aG sudo username`.
GRUB default OS	Edit `/etc/default/grub`, change `GRUB_DEFAULT`, then run `sudo update-grub`.

Next: Chapter 7 — Desktop environments. We explore GNOME, KDE Plasma, XFCE, MATE, and Cinnamon — what each looks like, how to install an alternative desktop, and how to switch between them at login.

Chapter 7 — Desktop Environments

One of the most distinctive things about Linux compared to Windows or macOS is that the desktop is not built into the operating system — it's a separate piece of software you can swap out. This layer is called the **desktop environment** (DE), and Linux offers several well-developed options. This chapter explains what each one looks like, how they compare, and how to install and switch between them.

> **Your distro already chose one.** When you installed Ubuntu, Mint, or Fedora, a desktop environment came with it. You don't need to install another one unless you want to explore or your current one doesn't suit you. This chapter is also useful for understanding what you have and why it was chosen.

1. What Is a Desktop Environment?

A desktop environment is a collection of software that together creates the graphical interface you interact with day-to-day. It includes:

- **Window manager** — draws and manages application windows; handles minimise, maximise, and moving windows around the screen
- **Panel / taskbar** — the bar(s) at the top or bottom showing open windows, system clock, and notification area
- **Application launcher** — the menu or search interface for starting applications
- **File manager** — the graphical tool for browsing files and folders
- **Settings application** — for configuring appearance, displays, sound, and so on
- **Bundled applications** — text editor, image viewer, calendar, and other utilities that come with the DE

Different DEs make very different choices about how all of these work — which is why switching from GNOME to KDE can feel almost like switching to a different operating

system, even though the underlying Linux is identical.

2. The Main Desktop Environments

GNOME

~700 MB RAM | Modern look | Extensions

Used by: Ubuntu, Fedora, Debian (default)

GNOME (GNU Network Object Model Environment) is the most widely used Linux desktop. Its design philosophy is "less is more" — it deliberately removes clutter, starting with no taskbar by default. The workflow centres on the **Activities Overview**, accessed by pressing the Super key (Windows key) or moving the mouse to the top-left corner. This shows all open windows, a search bar, and virtual workspaces.

GNOME looks clean and modern but takes some adjustment for users coming from Windows — there's no traditional Start menu. It shines on touchscreen laptops and makes heavy use of gestures on trackpads. Extensions (add-ons from extensions.gnome.org) can restore a traditional taskbar, add a dock, or fundamentally alter the workflow.

RAM USAGE (IDLE)	CUSTOMISATION	BEST FOR
~700–900 MB	Medium — via GNOME Tweaks and Extensions	Modern hardware, touchscreens, clean workflow

FILE MANAGER	DISPLAY MANAGER	LEARNING CURVE
Nautilus (Files)	GDM (GNOME Display Manager)	Medium — different workflow than Windows

KDE Plasma

Used by: KDE Neon, Kubuntu, Fedora KDE spin, openSUSE (default)

~500 MB RAM | Polished | Highly customisable

KDE Plasma is the most feature-rich and customisable Linux desktop. It has a traditional Windows-like layout out of the box — taskbar at the bottom, Start-menu-

style application launcher, system tray — but almost every element can be repositioned, resized, and restyled. Plasma supports widgets on the desktop, multiple panel layouts, and a global theme system that can transform its appearance completely.

Despite being feature-packed, Plasma is surprisingly lightweight — it uses less RAM than GNOME while offering far more configuration options. It also has the best multi-monitor support of any Linux desktop. Downside: the sheer number of settings can be overwhelming.

RAM USAGE (IDLE)	CUSTOMISATION	BEST FOR
~500–700 MB	Extremely high — almost everything is configurable	Windows switchers, power users, multi-monitor setups

FILE MANAGER	DISPLAY MANAGER	LEARNING CURVE
Dolphin	SDDM	Low — familiar layout, deep settings when ready

Cinnamon

~500 MB RAM | Windows-like | Good customisation

Used by: Linux Mint (default)

Cinnamon was created by the Linux Mint team as a response to GNOME 3's controversial redesign. It keeps the traditional desktop layout that Windows users expect: taskbar at the bottom, application menu in the bottom-left corner, system tray on the right. This familiarity makes it the most recommended desktop for Windows switchers.

Cinnamon is polished, stable, and well-integrated with Linux Mint's tools. It supports "applets" (panel add-ons), "desklets" (desktop widgets), and themes. It doesn't have the cutting-edge look of GNOME or the raw customisability of KDE, but it gets out of your way and just works.

RAM USAGE (IDLE)	CUSTOMISATION	BEST FOR
~500–600 MB	Good — themes, applets, desklets	Windows switchers, general everyday use

FILE MANAGER	DISPLAY MANAGER	LEARNING CURVE
Nemo	LightDM (MDM on older Mint)	Very low — immediately familiar from Windows

XFCE

~300 MB RAM | Traditional | Configurable

Used by: Xubuntu, Linux Mint XFCE, Manjaro XFCE

XFCE is the go-to desktop for older hardware and anyone who prioritises speed and stability over visual flash. It uses far less RAM and CPU than GNOME or KDE while still providing a complete, traditional desktop experience. It looks a little dated compared to the others but is rock-solid — XFCE rarely has dramatic changes between versions, which means you learn it once and it stays familiar.

XFCE's panel system is highly flexible — you can add, remove, and rearrange panels and their contents. It's also one of the easiest desktops to get working well on very old hardware with limited RAM.

RAM USAGE (IDLE)	CUSTOMISATION	BEST FOR
~300–400 MB	Good — panels, themes, compositing on/off	Old hardware, users who want speed over looks

FILE MANAGER	DISPLAY MANAGER	LEARNING CURVE
Thunar	LightDM	Low — traditional layout, minimal surprises

MATE

~350 MB RAM | Classic GNOME feel

Used by: Ubuntu MATE, Linux Mint MATE edition

MATE was created to continue the GNOME 2 desktop after GNOME moved to its controversial version 3 design. If you've seen screenshots of Linux from 2005–2010, you've seen what MATE looks like — two panels (one at top, one at bottom), a classic application menu, and a clean traditional aesthetic. It's lightweight, stable, and has excellent accessibility support.

MATE sits between XFCE and Cinnamon in terms of resource use and modernity. It's a good choice if you want something more traditional than Cinnamon but with more polish than XFCE.

RAM USAGE (IDLE)	CUSTOMISATION	BEST FOR
~350–450 MB	Good — panels, applets, themes	Older hardware, GNOME 2 nostalgia, accessibility

FILE MANAGER	DISPLAY MANAGER	LEARNING CURVE
Caja	LightDM	Very low — extremely traditional layout

> **There are many more.** Beyond these five, notable alternatives include **LXQt** (even lighter than XFCE — used by Lubuntu), **Budgie** (sleek and macOS-inspired — used by Solus and Ubuntu Budgie), **Pantheon** (macOS-like — used exclusively by elementary OS), and **i3 / Sway** (tiling window managers — no overlapping windows, fully keyboard-driven, loved by developers). Tiling managers are powerful but have a steep learning curve.

3. Display Managers — The Login Screen

The **display manager** (DM) is the program that shows you the login screen after boot. It handles authentication (checking your password) and then launches your chosen desktop environment. When you have multiple DEs installed, the display manager is where you choose which one to use for that session.

Display Manager	Used by default with	Character
GDM (GNOME Display Manager)	GNOME, Ubuntu, Fedora	Clean, modern, supports Wayland sessions. Slightly heavier than the others.
SDDM (Simple Desktop Display Manager)	KDE Plasma, KDE Neon, Kubuntu	QML-based (same toolkit as KDE); highly themeable; supports Wayland.

Display Manager	Used by default with	Character
LightDM	XFCE, MATE, Cinnamon, Xubuntu, Linux Mint	Lightweight, fast, works with many "greeter" themes. The most flexible choice when mixing DEs.
XDM (X Display Manager)	Minimal installs, some server setups	Very basic, text-like appearance. Rarely used on modern desktops.

> **You don't normally need to think about the display manager.** It comes installed with your desktop environment and just works. It only becomes relevant when you install a second desktop and need to choose between them at login — the display manager provides the session picker.

4. Installing a Second Desktop Environment

You can install multiple desktop environments on the same system and switch between them at the login screen. This is a great way to try alternatives without reinstalling Linux.

> **Installing a second DE can add clutter.** Each desktop brings its own set of applications — a second file manager, a second text editor, a second settings app. These coexist fine but the application menu can get crowded. If you decide you prefer the new DE, you can remove the original one later. If you just want to try it briefly, that's fine too.

Installing KDE Plasma on Ubuntu

```
# Install KDE Plasma desktop (minimal - just the core, fewer KDE apps)
$ sudo apt install kde-plasma-desktop

# Or the full KDE experience with all KDE applications:
$ sudo apt install kde-full
```

Installing XFCE on Ubuntu

```
$ sudo apt install xfce4 xfce4-goodies
```

Installing GNOME on Linux Mint

```
$ sudo apt install ubuntu-gnome-desktop
```

Installing MATE on Ubuntu

```
$ sudo apt install ubuntu-mate-desktop
```

Installing Cinnamon on Ubuntu

```
$ sudo apt install cinnamon-desktop-environment
```

After installing, **log out or restart** for the new desktop to appear as an option at the login screen.

5. Switching Between Desktops at Login

Once you have multiple desktops installed, selecting between them at the login screen is straightforward — but the exact method depends on which display manager you have.

GDM (Ubuntu's default)

1. At the login screen, click your username.
2. Look for a **gear icon (⚙)** near the "Sign In" button in the bottom-right area of the password field.
3. Click the gear and a menu appears listing all available desktop sessions (e.g., "Ubuntu", "Ubuntu on Xorg", "GNOME", "Plasma (Wayland)", "XFCE Session").
4. Select your desired desktop and enter your password. GDM remembers your last choice — next login it defaults to the same desktop.

SDDM (KDE's default)

SDDM shows a session selector (usually a dropdown) directly on the login screen, often in the bottom-left corner. Select the session type, then log in normally.

LightDM

LightDM's appearance varies by the "greeter" theme installed. On Linux Mint (Slick Greeter), there's a small icon or dropdown near the username showing available sessions. On Xubuntu, look for a session button in the corner of the login screen.

> **Wayland vs X11 sessions.** You may see entries like "Ubuntu" (Wayland) and "Ubuntu on Xorg" (X11). Wayland is the modern replacement for X11 — it's more secure and handles HiDPI better, but some older applications and screen-sharing tools don't yet fully support it. If something doesn't work as expected, try the Xorg session.

6. Removing a Desktop Environment

If you've tried a second desktop and decided you don't want it, you can remove it. Be careful — removing a DE also removes its associated applications, and on some configurations, removing the wrong package can break the remaining desktop. Always reboot and confirm your remaining desktop works before removing anything.

Example: removing KDE from Ubuntu

```
# Remove KDE and its dependencies (be sure Ubuntu GNOME is working first)
$ sudo apt remove kde-plasma-desktop kde-full
$ sudo apt autoremove
$ sudo apt install ubuntu-desktop  # Reinstall Ubuntu's defaults if anything was
```

> **If in doubt, don't remove.** Having an extra desktop installed uses disk space (typically 500 MB–2 GB) but otherwise causes no harm. It's better to leave an unused desktop installed than to accidentally break your primary one by removing the wrong packages.

7. Which Desktop Should You Use?

Coming from Windows?

Old or low-spec hardware?

Cinnamon (Linux Mint) or **KDE Plasma**. Both have a familiar taskbar-and-start-menu layout. Cinnamon is slightly simpler; KDE has more features.

XFCE is the standard recommendation. **MATE** is also good. Both run well with 1–2 GB RAM. LXQt (Lubuntu) is lighter still for very old machines.

Want maximum customisation?

KDE Plasma. Almost every visual and behavioural element can be changed — colours, fonts, window decorations, panel positions, workspace behaviour, animations.

Just want something that looks great and stays out of the way?

GNOME. Minimal, clean, and deliberately uncluttered. Takes adjustment but many users love it once they adapt to the workflow.

Using a touchscreen laptop?

GNOME has the best touchscreen and gesture support of any Linux desktop. KDE Plasma is catching up but GNOME is still ahead here.

Want stability above all else?

XFCE or **MATE**. Both change slowly and predictably. An XFCE install from three years ago looks and behaves almost identically to a fresh one today.

Stick with what your distro chose. Your distro's default DE has been carefully tested and integrated with the rest of the system. Unless you have a specific reason to change, give the default a fair trial — at least a few weeks — before switching. You'll usually find it grows on you.

Chapter Summary

Desktop	RAM (idle)	Best for	Distros
GNOME	~700–900 MB	Modern hardware, clean workflow, touchscreens	Ubuntu, Fedora, Debian
KDE Plasma	~500–700 MB	Windows switchers, heavy customisation, multi-monitor	Kubuntu, KDE Neon, Fedora KDE, openSUSE
Cinnamon	~500–600 MB	Windows switchers, everyday general use	Linux Mint (default)
XFCE	~300–400 MB	Old hardware, speed, stability	Xubuntu, Mint XFCE, Manjaro XFCE
MATE	~350–450 MB	Traditional layout, older hardware, accessibility	Ubuntu MATE, Mint MATE

Desktop	RAM (idle)	Best for	Distros
Switching	Install additional DE via `apt install`, log out, click session gear icon at login screen		
Display managers	GDM (GNOME), SDDM (KDE), LightDM (XFCE/MATE/Cinnamon) — manage the login screen and session selection		

Next: Chapter 8 — Configuring sound. We look at how Linux handles audio (PulseAudio vs PipeWire), the graphical sound settings, HDMI audio output, and how to troubleshoot the most common "no sound" problems.

Chapter 8 — Configuring Sound

Sound on Linux works well on most hardware straight out of the box — but when it doesn't, it can be confusing because the audio stack has several layers with unfamiliar names. This chapter explains how Linux handles audio, how to use the graphical sound settings, how to switch output devices (including HDMI and Bluetooth), and how to diagnose and fix the most common sound problems.

1. How Linux Handles Audio — The Stack

Linux audio goes through several layers between an application and your speakers. Understanding these layers makes troubleshooting much easier.

Applications
Firefox, VLC, Spotify, games — anything that plays or records audio. These send audio to the sound server.

Sound Server — PipeWire or PulseAudio
Manages multiple audio streams, mixing them together. Controls volume per application. Routes audio to the correct output device. This is what you configure in the GUI settings panel.

ALSA (Advanced Linux Sound Architecture)
The kernel-level audio driver layer. Talks directly to the hardware. You rarely need to touch this; the sound server sits on top of it.

Hardware — Sound Card / Chip
The physical audio hardware: built-in motherboard audio, USB DAC, HDMI audio via GPU, or Bluetooth adapter.

When you adjust the volume slider in your desktop's settings panel, you're controlling the **sound server**. When audio is completely absent, the problem is usually either at the ALSA level (wrong device selected, muted) or at the sound server level (service not running, wrong output selected).

2. PulseAudio vs PipeWire

There are two sound servers in common use on Linux. Which one you have depends on your distro and how recently it was released.

PulseAudio — the established standard

- Has been the default Linux sound server since ~2008
- Handles desktop audio mixing well
- Bluetooth audio support is good but requires separate modules
- Not designed for professional low-latency audio
- Still the default on older Ubuntu LTS, Debian, Linux Mint 20
- Stable and very well-tested

PipeWire — the modern replacement

- Default on Ubuntu 22.10+, Fedora 34+, Linux Mint 21+
- Drop-in replacement for PulseAudio — existing apps work unchanged
- Also replaces JACK (professional low-latency audio)
- Better Bluetooth codec support (AAC, aptX, LDAC)
- Lower latency — better for gaming and video calls
- Handles screen capture audio alongside video (important for Wayland)

> **Which one do you have?** Run this in a terminal: `pactl info | grep "Server Name"`. If the output says *PulseAudio (on PipeWire)* you have PipeWire with PulseAudio compatibility. If it just says *PulseAudio*, you have the original PulseAudio. Either way, the graphical controls and most of the commands in this chapter work identically for both.

Checking what's running

```
$ pactl info | grep "Server Name"
Server Name: PulseAudio (on PipeWire 1.0.1)

# Or check which services are active:
$ systemctl --user status pipewire pulseaudio
```

3. Graphical Sound Settings

For everyday audio management you rarely need the terminal. Every major desktop environment has graphical sound controls.

Ubuntu (GNOME) — Settings → Sound

Open **Settings → Sound**. You'll find:

- **Output device** — dropdown showing all available audio outputs (built-in speakers, headphone jack, HDMI, USB audio). Select the one you want to use.
- **Output volume** — the master volume slider.
- **Input device** — microphone selection and input volume.
- **Alert volume** — volume for system notification sounds.
- **Test** button — plays a short sound to confirm the selected output works.

Linux Mint (Cinnamon) — Sound applet

Right-click the speaker icon in the system tray and choose **Sound Settings**. Alternatively, open **System Settings → Sound**. Mint's sound panel is similar to GNOME's but also shows per-application volume levels on the "Applications" tab.

KDE Plasma — Audio Volume widget

Click the speaker icon in the system tray to open the quick volume control. For full settings, right-click the speaker and choose **Configure Audio Devices** (or open **System Settings → Audio**). KDE lets you set a default output per application — for example, routing game audio to headphones and notification sounds to speakers.

XFCE / MATE — PulseAudio Volume Control

These desktops use a standalone tool. Search for **PulseAudio Volume Control** (pavucontrol) in the application menu. If it's not installed:

```
$ sudo apt install pavucontrol
```

pavucontrol is actually the most powerful graphical audio tool on Linux regardless of desktop — it shows every audio stream, every device, and lets you reroute streams between devices in real time.

4. pavucontrol — The Audio Power Tool

PulseAudio Volume Control (pavucontrol) works with both PulseAudio and PipeWire and gives you far more control than the basic desktop sound panel. It has five tabs:

- **Playback** — shows every application currently playing audio, with an individual volume slider for each. You can mute a specific app or adjust its volume independently of the master.
- **Recording** — shows every application currently recording audio (mic input). Useful for identifying which app has grabbed the microphone.
- **Output Devices** — lists every audio output (speaker, headphones, HDMI, USB). You can set the default and adjust the volume for each device independently.
- **Input Devices** — lists every microphone and audio input. Set the default and adjust input gain.
- **Configuration** — shows the profile for each audio card. This is where you switch between stereo output, surround sound, HDMI audio, and digital output (S/PDIF). **This tab fixes most HDMI audio problems.**

> **pavucontrol is the first tool to open when sound isn't working.** The Playback tab will show you whether audio is actually being sent to the sound server (the stream will appear, even if you can't hear it), and the Output Devices tab will show if the volume is muted at the device level rather than the application level.

5. HDMI Audio

HDMI carries audio as well as video, so when you connect a monitor or TV via HDMI you expect audio to come from those speakers. Linux handles this but sometimes needs a nudge to select the right output.

1. Open **pavucontrol** (or Settings → Sound on GNOME).
2. Click the **Configuration** tab. You'll see your audio hardware listed — typically something like "Built-in Audio" and possibly a separate GPU audio device (e.g., "NVIDIA High Definition Audio" or "AMD HD Audio").

3. For the GPU audio device, change its profile from "Off" to **"Digital Stereo (HDMI)"** or the matching HDMI port (HDMI 1, HDMI 2, etc.).
4. Go to the **Output Devices** tab — the HDMI output should now appear. Click the green tick (✓) button next to it to set it as the default.
5. Test by playing any audio. If using GNOME, you can also set the output in Settings → Sound → Output device dropdown.

> **Multiple HDMI ports:** if your GPU has more than one HDMI/DisplayPort output, each may appear as a separate audio device (HDMI 1, HDMI 2, HDMI 3…). If you're not sure which port you're using, try each profile in turn — only the one connected to a display with speakers will produce sound.

6. Bluetooth Audio

Bluetooth headphones and speakers work on Linux, though the initial pairing process is slightly different from Windows or macOS.

Pairing via the graphical interface

1. Put your Bluetooth device into pairing mode.
2. Open **Settings → Bluetooth** (Ubuntu/GNOME) or the Bluetooth manager from the system tray. Make sure Bluetooth is enabled.
3. Your device should appear in the "Available devices" list. Click it and then click **Pair**. Confirm any PIN if prompted.
4. After pairing, the Bluetooth headphones should appear in Settings → Sound → Output device. Select them.

Bluetooth audio quality — A2DP profile

Bluetooth audio has two main profiles: **A2DP** (high-quality stereo audio) and **HSP/HFP** (low-quality "headset" profile that enables the microphone). Linux sometimes defaults to HSP/HFP because it detects a microphone in the headphones. If your Bluetooth audio sounds terrible, this is why.

Fix it in pavucontrol → **Configuration** tab → select your Bluetooth device → change profile to **"High Fidelity Playback (A2DP Sink)"**.

> **PipeWire has better Bluetooth codec support** than PulseAudio, including AAC, aptX, and LDAC (high-quality audio compression used by Sony headphones). If you're on an older distro with PulseAudio and Bluetooth audio quality is poor, upgrading to PipeWire (or upgrading your distro) can make a noticeable difference.

7. Terminal Audio Tools

When the GUI isn't enough — or when you're troubleshooting over SSH on a headless system — the terminal provides complete control over audio.

List output devices

```
$ pactl list sinks short
0  alsa_output.pci-0000_00_1f.3.analog-stereo  PipeWire  s32le 2ch 48000Hz  IDLE
1  alsa_output.pci-0000_01_00.1.hdmi-stereo    PipeWire  s32le 2ch 48000Hz  SUSPE
```

Set the default output device

```
# Use the sink name from the list above
$ pactl set-default-sink alsa_output.pci-0000_01_00.1.hdmi-stereo
```

Adjust volume from the terminal

```
# Set master volume to 80%
$ pactl set-sink-volume @DEFAULT_SINK@ 80%

# Increase volume by 5%
$ pactl set-sink-volume @DEFAULT_SINK@ +5%

# Mute / unmute
$ pactl set-sink-mute @DEFAULT_SINK@ toggle
```

Check ALSA — the layer below PulseAudio/PipeWire

```
# List ALSA playback devices
$ aplay -l
**** List of PLAYBACK Hardware Devices ****
card 0: PCH [HDA Intel PCH], device 0: ALC295 Analog [ALC295 Analog]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 1: NVidia [HDA NVidia], device 3: HDMI 0 [HDMI 0]
  Subdevices: 1/1
  Subdevice #0: subdevice #0

# Play a test tone directly via ALSA (bypasses PulseAudio/PipeWire)
$ speaker-test -c 2 -t wav
```

> `speaker-test` bypasses PipeWire/PulseAudio and talks directly to ALSA. If you hear sound from `speaker-test` but not from applications, the problem is in the sound server layer. If `speaker-test` is also silent, the problem is at the ALSA/driver/hardware level.

8. Troubleshooting Common Sound Problems

No sound at all after a fresh install

1. Check the volume isn't muted. Open pavucontrol → Output Devices and check that the volume slider isn't at zero and the mute button isn't active. Also check the Playback tab — if no streams appear, the application might not be sending audio at all.

2. Check the ALSA mixer isn't muted. Open a terminal and run `alsamixer`. Use arrow keys to navigate; press **M** to unmute a channel (muted channels show "MM" at the bottom; unmuted show "OO"). Press F6 to select a different sound card if you have more than one.

Sound works on headphones but not on speakers (or vice versa)

Headphone jack detection: Linux sometimes fails to switch automatically between speakers and headphones when you plug/unplug. Open `alsamixer`, press F6, select your audio card, and look for a "Headphone" or "Speaker" channel. Make sure the correct one isn't muted. Alternatively, in pavucontrol → Configuration, try switching the profile (e.g., from "Analog Stereo Output" to "Analog Stereo Duplex").

HDMI audio not working

Open pavucontrol → **Configuration** tab. Find the GPU audio device and change its profile from "Off" to "Digital Stereo (HDMI)". Then in the Output Devices tab, set the HDMI output as default. If multiple HDMI entries appear, try each one. Also confirm your monitor or TV has speakers and they're not muted on the TV's own controls.

Sound server crashed — no audio at all, pavucontrol won't open

Restart the sound server without logging out:

`systemctl --user restart pipewire pipewire-pulse` (PipeWire systems)
`pulseaudio -k && pulseaudio --start` (PulseAudio systems)

If the service keeps crashing, check for errors with `journalctl --user -u pipewire -n 50`.

Bluetooth audio keeps switching to bad quality (HSP/HFP)

This happens when an application activates the microphone on your Bluetooth headset. Open pavucontrol → **Configuration** tab, find your Bluetooth device, and manually set the profile to **"High Fidelity Playback (A2DP Sink)"**. This persists until another application activates the microphone again. For a permanent fix, configure your video-calling application to use a different microphone (e.g., built-in laptop mic).

Crackling, popping, or stuttering audio

Usually a buffer or timing problem. For PulseAudio: edit `/etc/pulse/default.pa` and find the line loading `module-udev-detect` — add `tsched=0` to disable timer-based scheduling:
`load-module module-udev-detect tsched=0`
Then restart PulseAudio. For PipeWire, try increasing the quantum (buffer size) in `~/.config/pipewire/pipewire.conf.d/` — but this is rarely needed on modern hardware.

Microphone not working or not detected

Open pavucontrol → **Input Devices** tab. If your microphone appears but has zero input level, the gain may be at zero — raise the slider. If it doesn't appear at all, check alsamixer (F4 for capture channels) to ensure the microphone input isn't muted. For USB microphones: unplug and re-plug, then check pavucontrol Input Devices — USB mics usually appear immediately without any driver installation.

Chapter Summary

Topic	Key points
Audio stack	Applications → Sound server (PipeWire/PulseAudio) → ALSA → Hardware. Graphical controls manage the sound server layer.

Topic	Key points
PipeWire vs PulseAudio	PipeWire is the modern default (Ubuntu 22.10+, Fedora 34+, Mint 21+). PulseAudio is on older installs. Both are controlled identically via `pactl` and pavucontrol.
Graphical control	GNOME: Settings → Sound. KDE: system tray speaker. XFCE/MATE: pavucontrol. pavucontrol works everywhere and is the most powerful option.
HDMI audio	Open pavucontrol → Configuration tab → enable HDMI profile on GPU audio device → set as default in Output Devices tab.
Bluetooth audio	Pair via Bluetooth settings. If quality is poor, set A2DP profile in pavucontrol → Configuration. PipeWire has better Bluetooth codec support.
Terminal tools	`pactl list sinks short` — list outputs. `pactl set-default-sink` — set default. `alsamixer` — check for muted ALSA channels. `speaker-test` — test ALSA directly.
No sound fix order	1) Check not muted in pavucontrol. 2) Check alsamixer for muted channels. 3) Check correct output selected. 4) Restart sound server. 5) Run speaker-test to isolate ALSA vs server problem.

Next: Chapter 9 — Configuring networking. We cover wired Ethernet, connecting to WiFi with NetworkManager, setting a static IP address, and troubleshooting common connection problems.

Chapter 9 — Configuring Networking

Networking on Linux is managed by a tool called **NetworkManager**, which handles both wired and wireless connections automatically on most desktop distros. For the majority of users, network setup is invisible — you plug in an Ethernet cable or click a WiFi network and it just works. This chapter covers the graphical tools, how to configure a static IP address, and what to do when things don't connect as expected.

1. How Linux Networking Is Organised

Like audio, networking in Linux has several layers. You don't need to understand them all in depth, but knowing which layer a problem is at saves a lot of troubleshooting time.

Applications — Browser, email, SSH, etc.
These use the network through standard system calls. They don't manage connections themselves.

NetworkManager — connection management
Manages which networks you're connected to, handles DHCP, saves WiFi passwords, creates VPN connections. This is what you configure in the GUI settings panel.

Linux kernel — network stack and drivers
TCP/IP implementation, routing tables, firewall (iptables/nftables). Network interface drivers live here. You interact with this layer via ip, ss, and ping commands.

Hardware — Ethernet NIC, WiFi adapter
The physical network interface card or WiFi chip. Requires a matching kernel driver to function.

> **Server distros use different tools.** Ubuntu Server and Debian use **systemd-networkd** and **Netplan** instead of NetworkManager for network configuration. This chapter focuses on desktop setups where NetworkManager is the standard. Server networking is a topic for a later, more advanced course.

2. Wired Ethernet

Ethernet is the simplest case. Plug in a cable and NetworkManager connects automatically — it requests an IP address via DHCP from your router and you're online within seconds. No configuration is needed for most home and office setups.

Verify your Ethernet connection

The network icon in your system tray will show a plug symbol when Ethernet is connected. On GNOME, click the top-right corner and look for the wired network status. On Mint/Cinnamon, the network applet in the taskbar shows connection status.

```
# Check all network interfaces and their status
$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 ...
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 ...
    link/ether a0:b1:c2:d3:e4:f5 brd ff:ff:ff:ff:ff:ff
3: wlp2s0: <BROADCAST,MULTICAST> mtu 1500 ...
```

Interface names follow a predictable pattern on modern Linux:

- **enp3s0 / ens33 / etho** — Ethernet (wired) interfaces. `en` = Ethernet, `p3s0` = PCI bus location.
- **wlp2s0 / wlano** — WiFi (wireless) interfaces. `wl` = wireless LAN.
- **lo** — the loopback interface (127.0.0.1), always present, always up.

Check your IP address

```
$ ip addr show
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP>
    inet 192.168.1.105/24 brd 192.168.1.255 scope global dynamic enp3s0
       valid_lft 85432sec preferred_lft 85432sec

# Or shorter — just show the IP addresses:
$ hostname -I
192.168.1.105
```

3. WiFi — Connecting Graphically

WiFi connection on a Linux desktop is nearly identical to Windows. NetworkManager handles scanning, connecting, and saving passwords automatically.

Ubuntu (GNOME)

1. Click the **top-right corner** of the screen to open the quick settings panel. Click the WiFi icon or the arrow next to it.
2. A list of available networks appears. Click your network name.
3. Enter the WiFi password when prompted and click **Connect**. NetworkManager saves the password — you won't need to enter it again.

Linux Mint (Cinnamon) / XFCE / MATE

Click the **network icon** in the system tray (bottom-right of the screen). Available WiFi networks are listed. Click one, enter the password, and connect. The process is identical in behaviour to Windows.

Managing saved connections

To view, edit, or delete saved WiFi connections, right-click the network icon and choose **"Edit Connections"** (or **Network Connections** on Mint). This opens the NetworkManager connection editor where you can see all saved networks, change passwords, set auto-connect behaviour, and configure advanced options.

> **WiFi not appearing?** First check that the WiFi adapter isn't turned off. Look for a hardware switch on the laptop (many have a physical WiFi toggle key, often Fn+F2 or similar). Also check Settings → WiFi to make sure the software toggle is on. If the adapter doesn't appear at all, it may need a proprietary driver — see Chapter 6 on Additional Drivers.

4. WiFi — Connecting from the Terminal

If you need to connect to WiFi without a graphical interface (a server install, a broken desktop, or an SSH session), use **nmcli** — the NetworkManager command-line tool.

```
# List all visible WiFi networks
$ nmcli device wifi list
IN-USE  SSID              MODE   CHAN  RATE        SIGNAL  SECURITY
        MyHomeNetwork     Infra  6     130 Mbit/s  87      WPA2
        Neighbour_5G      Infra  36    405 Mbit/s  52      WPA2

# Connect to a network (will prompt for password)
$ nmcli device wifi connect "MyHomeNetwork" password "mysecretpassword"
Device 'wlp2s0' successfully activated with '...'

# Show all saved/active connections
$ nmcli connection show

# Disconnect from current WiFi
$ nmcli device disconnect wlp2s0

# Reconnect to a saved network by name
$ nmcli connection up "MyHomeNetwork"
```

5. Checking Connectivity from the Terminal

A handful of terminal commands are invaluable for diagnosing network problems. These work on every Linux distro and are worth knowing even if you rarely use the terminal.

Command	What it does	Example
`ping`	Sends packets to a host and reports if they arrive. Tests basic connectivity.	`ping -c 4 google.com`
`ip addr`	Shows IP addresses assigned to all network interfaces.	`ip addr show enp3s0`
`ip route`	Shows the routing table — how traffic is directed to destinations including the default gateway.	`ip route show`
`nmcli`	NetworkManager CLI — manage connections, check status, connect to WiFi.	`nmcli device status`

Command	What it does	Example
`ss`	Shows open network sockets — which ports are listening and which connections are active.	`ss -tuln`
`dig / nslookup`	DNS lookup — resolves a hostname to an IP address. Tests DNS is working.	`dig google.com`
`curl`	Makes HTTP/HTTPS requests. Quick test for internet connectivity and web server responses.	`curl -I https://google.com`
`traceroute`	Shows the path packets take to reach a destination — identifies where a connection fails.	`traceroute google.com`

A quick connectivity test sequence

```
# 1. Is the interface up and do I have an IP address?
$ ip addr show
# Look for "inet x.x.x.x" under your interface — if missing, no DHCP lease

# 2. Can I reach my router (replace with your gateway IP)?
$ ping -c 3 192.168.1.1
# If this fails: physical connection problem or wrong gateway

# 3. Can I reach the internet by IP (bypasses DNS)?
$ ping -c 3 8.8.8.8
# If this fails but step 2 works: router/ISP problem

# 4. Can I resolve domain names (tests DNS)?
$ ping -c 3 google.com
# If this fails but step 3 works: DNS problem
```

6. Setting a Static IP Address

By default, your computer gets an IP address from your router via DHCP — the address can change each time you reconnect. For home servers, printers, or any device you want to reach by a fixed address, a static IP is useful.

> **Router reservation vs static IP on the device.** The easiest approach for home use is to configure a **DHCP reservation** in your router settings — the router always gives the same IP to your computer's MAC address. This avoids configuring anything on the Linux side. Static IP configuration on the device itself is useful when you don't control the router (office networks, VPSes) or need an address outside the DHCP range.

Graphical method — NetworkManager

1. Click the network icon in the system tray → click the gear icon next to your active connection, or open **Settings → Network** and click the gear next to your connection.
2. Go to the **IPv4** tab.
3. Change the method from **"Automatic (DHCP)"** to **"Manual"**.
4. Enter your settings. Here's an example for a typical home network:

EXAMPLE STATIC IP CONFIGURATION — HOME NETWORK (192.168.1.X)

Address	`192.168.1.50`	Choose an address outside your router's DHCP range to avoid conflicts. Check your router settings for the DHCP range — often .100–.200, so .50 is safe.
Netmask	`255.255.255.0`	Also entered as /24. Standard for home networks.
Gateway	`192.168.1.1`	Your router's IP address. Run `ip route | grep default` to find it if unsure.
DNS	`1.1.1.1, 8.8.8.8`	Cloudflare (1.1.1.1) and Google (8.8.8.8) are reliable public DNS servers. Or use your router's IP as DNS (it forwards to your ISP).

5. Click **Apply**, then disconnect and reconnect the network connection for the static IP to take effect.

Terminal method — nmcli

```
# Find your connection name first
$ nmcli connection show
NAME            UUID                  TYPE      DEVICE
Wired           a1b2c3...             ethernet  enp3s0
MyHomeNetwork   d4e5f6...             wifi      wlp2s0
```

```
# Set a static IP on the wired connection
$ nmcli connection modify "Wired" \
    ipv4.method manual \
    ipv4.addresses 192.168.1.50/24 \
    ipv4.gateway 192.168.1.1 \
    ipv4.dns "1.1.1.1,8.8.8.8"

# Apply the change by bouncing the connection
$ nmcli connection down "Wired" && nmcli connection up "Wired"

# Revert to DHCP if needed
$ nmcli connection modify "Wired" ipv4.method auto ipv4.addresses "" ipv4.gateway
$ nmcli connection down "Wired" && nmcli connection up "Wired"
```

7. DNS Configuration

DNS (Domain Name System) translates hostnames like `google.com` into IP addresses. On most desktop installations, NetworkManager handles DNS automatically using your router's DNS server. You can override this with faster or more privacy-respecting public DNS servers.

Check current DNS servers

```
$ resolvectl status
Global
       Protocols: -LLMNR -mDNS -DNSOverTLS DNSSEC=no/unsupported
resolv.conf mode: stub

Link 2 (enp3s0)
    Current Scopes: DNS
         Protocols: +DefaultRoute
Current DNS Server: 192.168.1.1
       DNS Servers: 192.168.1.1
```

Popular public DNS servers

- **Cloudflare** — `1.1.1.1` and `1.0.0.1` — fast, privacy-focused, no logging
- **Google** — `8.8.8.8` and `8.8.4.4` — very reliable, widely used
- **Quad9** — `9.9.9.9` — blocks known malicious domains automatically

Set DNS via NetworkManager as shown in the static IP section above, or in the graphical IPv4 settings panel of your connection.

8. Troubleshooting Common Network Problems

No internet after connecting to WiFi — shows "connected" but nothing loads

Run the 4-step connectivity test from Section 5. If you can ping `8.8.8.8` but not `google.com`, the problem is DNS. Try `ping 1.1.1.1` — if that works too, set a manual DNS server via NetworkManager (1.1.1.1 or 8.8.8.8). Restart the connection after changing DNS. If `8.8.8.8` also fails, the problem is upstream of your machine — check your router.

WiFi disconnects frequently or has an unstable signal

Power management: Linux WiFi drivers sometimes aggressively power-down the adapter to save battery, causing dropouts. Disable WiFi power management:
`sudo iwconfig wlp2s0 power off`
To make this permanent, create `/etc/NetworkManager/conf.d/wifi-powersave-off.conf` containing:
`[connection]`
`wifi.powersave = 2`
Then restart NetworkManager: `sudo systemctl restart NetworkManager`.

WiFi adapter not detected — no networks appear at all

1. Check `ip link show` — if no `wl` interface appears, the driver isn't loaded. Run `lspci | grep -i wireless` or `lsusb | grep -i wireless` to identify the chip. Search for "[chip name] Linux driver" — Broadcom and some Realtek chips require proprietary packages.

2. Try the Ubuntu/Mint Additional Drivers tool (Chapter 6). It detects and installs Broadcom drivers automatically if connected via Ethernet.

Ethernet not working — cable plugged in but no connection

Check `ip link show` — the Ethernet interface should show `UP` and `LOWER_UP` when a cable is connected. If it shows `DOWN`, the cable or port may be faulty — try a different cable and port.
If `UP` but no IP address: `sudo dhclient enp3s0` forces a new DHCP request. If that fails, check your router's DHCP server is enabled.

Can't connect to a specific WiFi network (wrong password error even with correct password)

Delete the saved connection and reconnect from scratch. In NetworkManager connection editor, find the network in the saved list and delete it. Or via terminal: `nmcli connection delete "NetworkName"`. Then reconnect as if for the first time and re-enter the password carefully — pay attention to case sensitivity.

NetworkManager not running — nmcli gives "Error: NetworkManager is not running"

Start and enable NetworkManager:
`sudo systemctl start NetworkManager`
`sudo systemctl enable NetworkManager`
If it fails to start, check the logs: `sudo journalctl -u NetworkManager -n 50`. On server distros that use Netplan/networkd instead, NetworkManager may be intentionally absent.

Chapter Summary

Topic	Key points
NetworkManager	Manages all connections on desktop Linux. Handles DHCP, saves WiFi passwords, creates VPNs. Configured via GUI tray icon or `nmcli` in terminal.
Ethernet	Auto-connects on cable insertion. Check with `ip link show` (interface should be UP) and `ip addr show` (should have an inet address).
WiFi (GUI)	Click network tray icon → select network → enter password. NetworkManager saves the password and reconnects automatically.
WiFi (terminal)	`nmcli device wifi list` to scan, `nmcli device wifi connect "Name" password "pass"` to connect.
Connectivity test	Ping gateway → ping 8.8.8.8 → ping google.com. Failure at each step tells you which layer the problem is at.
Static IP	Set via NetworkManager GUI (IPv4 → Manual) or `nmcli connection modify`. Prefer router DHCP reservation for home use.
DNS	Managed automatically by NetworkManager. Override with 1.1.1.1 (Cloudflare), 8.8.8.8 (Google), or 9.9.9.9 (Quad9) for speed or privacy.
WiFi dropout fix	Disable power management: create `/etc/NetworkManager/conf.d/wifi-powersave-off.conf` with `wifi.powersave = 2`.

Next: Chapter 10 — Installing software. We cover the Software Centre graphical tool, the basics of apt and dnf/yum, and the universal package formats Flatpak, Snap, and AppImage.

Chapter 10 — Installing Software

One of the most common questions from new Linux users is: "how do I install software?" The answer is quite different from Windows or macOS — and in many ways considerably easier. Linux doesn't involve hunting for installers on random websites, running unsigned .exe files, or clicking through wizard screens. Software comes from **repositories** — curated, verified collections maintained by your distro — and a package manager handles downloading, installing, and updating everything.

> **The Linux approach to software.** Think of the package manager as an app store that covers your entire system — not just apps, but system tools, fonts, libraries, drivers, and everything else. It tracks what's installed, resolves dependencies automatically, and can update everything with a single command. Most software you'll ever need is in the repositories.

1. The Graphical Software Centre

Every major desktop distro ships a graphical app store. You can browse, search, install, and remove software entirely by clicking — no terminal required.

Ubuntu — GNOME Software

Open **Ubuntu Software** (the shopping bag icon) from the applications menu. Search for an application by name, click it, and click **Install**. You'll be asked for your password once. The application appears in your applications menu when done.

> **Snap packages in Ubuntu Software.** Ubuntu's Software Centre installs many applications as Snaps by default — even when a traditional `.deb` package is available. Snaps are larger and slower to start than native packages. If startup speed matters, use the terminal with `apt install` to get the native package instead.

Linux Mint — Software Manager

Mint's **Software Manager** is widely regarded as the best graphical package manager on Linux. Open it from the taskbar or application menu. It shows user ratings, screenshots, and clear descriptions. Searches are fast and results come from Mint's repositories (traditional .deb packages, not Snaps). Click **Install** and enter your password.

Fedora — GNOME Software

Fedora uses GNOME Software, which draws from Fedora's RPM repositories and Flathub (the main Flatpak repository). Flatpak is Fedora's preferred format for third-party applications. Install the same way — search and click Install.

KDE Discover

KDE-based distros (Kubuntu, KDE Neon, openSUSE with KDE) use **Discover**, KDE's software centre. It supports both native packages and Flatpaks and has a similar browse-and-click interface.

2. apt — The Debian/Ubuntu Package Manager

apt (Advanced Package Tool) is the package manager for all Debian-based distros — Ubuntu, Linux Mint, Pop!_OS, Zorin OS, and many others. It's the command you'll see in most online guides for Ubuntu-based systems. You need `sudo` for any command that modifies the system.

The essential apt commands

```
# Update the package list (always do this before installing)
$ sudo apt update

# Install a package
$ sudo apt install vlc

# Install multiple packages at once
$ sudo apt install vlc gimp inkscape

# Search for a package by name or description
$ apt search "video player"
```

```
# Show details about a package before installing
$ apt show vlc

# Remove a package (keeps config files)
$ sudo apt remove vlc

# Remove a package AND its config files
$ sudo apt purge vlc

# Remove packages that were auto-installed and are no longer needed
$ sudo apt autoremove

# Update all installed packages
$ sudo apt upgrade

# Update package list AND upgrade everything in one command
$ sudo apt update && sudo apt upgrade -y
```

apt quick reference

Command	What it does
`sudo apt update`	Refresh the list of available packages from repositories. Run before installing.
`sudo apt install <pkg>`	Download and install a package and all its dependencies.
`sudo apt remove <pkg>`	Uninstall a package, keeping configuration files.
`sudo apt purge <pkg>`	Uninstall a package and delete its configuration files.
`sudo apt upgrade`	Install available updates for all installed packages.
`sudo apt autoremove`	Remove orphaned packages no longer needed as dependencies.
`apt search <term>`	Search package names and descriptions.
`apt show <pkg>`	Display package details: description, size, dependencies, version.
`dpkg -l | grep <name>`	List installed packages matching a name.

> **Always run** `sudo apt update` **before** `sudo apt install`**.** Without it, apt uses a cached package list that may be days or weeks old, and will either install an old version or fail to find the

package entirely. It takes only a few seconds.

What are repositories and PPAs?

A **repository** (repo) is a server hosting a curated collection of packages. Your distro's official repositories contain thousands of tested, safe packages. Sometimes a package isn't in the official repos — perhaps it's too new, too niche, or proprietary. In those cases you can add a **PPA** (Personal Package Archive) — a third-party repository maintained by a developer or team.

```
# Example: add a PPA (third-party repository)
$ sudo add-apt-repository ppa:libreoffice/ppa
$ sudo apt update
$ sudo apt install libreoffice
```

Be careful with PPAs. Official repositories are maintained by your distro team and security-audited. PPAs are maintained by individuals — the quality and safety varies. Only add PPAs from sources you trust (official project pages, well-known developers). Avoid random PPAs found in forum posts.

3. dnf / yum — The Fedora/Red Hat Package Manager

dnf (Dandified YUM) is the package manager for Fedora, Red Hat Enterprise Linux, AlmaLinux, Rocky Linux, and CentOS. It replaced the older **yum** command — on modern systems `yum` is either an alias for `dnf` or not present. The concepts are identical to `apt`; only the syntax differs slightly.

```
# Install a package (also updates the package list automatically)
$ sudo dnf install vlc

# Search for a package
$ dnf search "video player"

# Show package details
$ dnf info vlc
```

```
# Remove a package
$ sudo dnf remove vlc

# Update all packages
$ sudo dnf upgrade

# Update package list and upgrade (--refresh forces a metadata refresh)
$ sudo dnf upgrade --refresh

# Remove unused dependencies
$ sudo dnf autoremove

# List installed packages
$ dnf list installed
```

> **dnf vs apt differences:** `dnf install` automatically refreshes the package list before installing — you don't need a separate `dnf update` step first (though `dnf upgrade --refresh` forces it). The command for updates is `dnf upgrade` not `dnf update` (though both work on modern dnf).

4. Universal Package Formats

Native packages (deb, rpm) are tied to a specific distro family. Three newer formats aim to work on *any* Linux distro — solving the problem of software that isn't in your distro's repositories.

Flatpak

RECOMMENDED UNIVERSAL FORMAT

- Works on any distro
- Main source: Flathub (flathub.org)
- Runs in a sandbox — isolated from the system
- Apps update independently of the OS
- Slightly larger disk footprint (shared runtimes)

Snap

UBUNTU'S UNIVERSAL FORMAT

- Developed by Canonical (Ubuntu's maker)
- Source: Snap Store (snapcraft.io)
- Also sandboxed and self-contained
- Slower startup than Flatpak or native packages
- Larger package sizes

AppImage

SINGLE-FILE PORTABLE APPS

- One file = one complete application
- No installation needed — just download and run
- No package manager or root access required
- No automatic updates (check manually)
- Good for trying software without installing

- Supported by Fedora, Mint, Pop!_OS by default
- Best for: desktop GUI apps not in repos

- Required for some Ubuntu defaults (Firefox)
- Linux Mint blocks Snaps by default

- Popular with audio/video production tools
- Best for: occasional-use or portable apps

5. Flatpak in Detail

Flatpak is the recommended universal format for most users. **Flathub** (flathub.org) is the central repository with thousands of applications — including many that aren't in distro repositories or are more up-to-date there than in official repos.

Setting up Flatpak (Ubuntu)

Flatpak is pre-installed on Fedora and Linux Mint. On Ubuntu it needs a one-time setup:

```
# Install Flatpak
$ sudo apt install flatpak

# Add the Flathub repository
$ flatpak remote-add --if-not-exists flathub https://dl.flathub.org/repo/flathub.

# Restart is recommended after setup
```

Installing and managing Flatpaks

```
# Search for an app on Flathub
$ flatpak search vlc

# Install from Flathub (use the app ID shown in search results)
$ flatpak install flathub org.videolan.VLC

# Run a Flatpak app
$ flatpak run org.videolan.VLC

# List installed Flatpak apps
$ flatpak list

# Update all Flatpak apps
```

```
$ flatpak update

# Remove a Flatpak app
$ flatpak uninstall org.videolan.VLC

# Remove unused Flatpak runtimes to free disk space
$ flatpak uninstall --unused
```

> **Flatpaks appear in your app launcher automatically.** After installing a Flatpak you don't need to use the terminal to launch it — it appears in the GNOME Activities, Mint application menu, or KDE application launcher just like any native application.

6. Snap in Detail

Snaps are pre-installed on Ubuntu. The `snap` command manages them from the terminal, and many also appear in the Ubuntu Software Centre.

```
# Search for a Snap
$ snap find vlc

# Install a Snap
$ sudo snap install vlc

# List installed Snaps
$ snap list

# Update all Snaps (Snaps also auto-update in the background)
$ sudo snap refresh

# Remove a Snap
$ sudo snap remove vlc
```

> **Snaps auto-update silently** in the background, even while you're using the app. This is convenient but can occasionally cause issues — an update mid-session may close a running application. You can delay this with `sudo snap set system refresh.hold="$(date --date='next month' +%Y-%m-%dT%H:%M:%S%:z)"` if it becomes a nuisance.

7. AppImage in Detail

AppImages work differently from Flatpak and Snap — there's no package manager involved at all. You download a single `.AppImage` file, make it executable, and run it.

1. **Download** the `.AppImage` file from the application's website.
2. **Make it executable.** Right-click the file in the file manager → Properties → Permissions → tick "Allow executing file as program". Or in the terminal:

```
$ chmod +x ~/Downloads/SomeApp-x86_64.AppImage
```

3. **Run it** by double-clicking in the file manager, or in the terminal:

```
$ ./SomeApp-x86_64.AppImage
```

To remove an AppImage, simply delete the file. No uninstaller, no package manager — it leaves nothing behind.

> **AppImages don't update automatically.** You need to check the application's website manually for new versions and download a fresh file. Some AppImages include a built-in update tool (`--appimage-update`), but many don't. For software you use regularly, Flatpak is more convenient.

> **FUSE requirement.** AppImages use FUSE (Filesystem in Userspace) to mount themselves. On Ubuntu 22.04+ and some other distros, FUSE 2 may not be installed. If an AppImage fails to run with a "FUSE" error: `sudo apt install libfuse2`.

8. Installing .deb and .rpm Files Directly

Some software (Google Chrome, VS Code, Discord, Slack) is distributed as a downloadable `.deb` or `.rpm` file from the developer's website rather than through a repository. This is similar to downloading an installer on Windows.

Installing a .deb file (Ubuntu/Mint)

```
# Double-click the .deb in the file manager (opens Software Centre)
# Or install via terminal — preferred as it also handles dependencies:
$ sudo apt install ./google-chrome-stable_current_amd64.deb
```

Using `apt install ./filename.deb` (note the `./`) is better than `dpkg -i` because apt automatically installs any missing dependencies.

Installing a .rpm file (Fedora)

```
$ sudo dnf install ./google-chrome-stable_current_x86_64.rpm
```

> **Many .deb/.rpm installers add a repository** so future updates come automatically via `apt upgrade` or `dnf upgrade`. Google Chrome and VS Code both do this — after the first install you never need to download the file again; updates arrive through the package manager.

9. Which Format Should You Use?

First choice:	**Native package (apt / dnf)** — always prefer the distro's own repository if the software is there. Native packages are the best integrated, fastest, and smallest.
Not in repos?	**Flatpak from Flathub** — the most widely supported universal format. Works on all distros, sandboxed, auto-updates. Check flathub.org before trying anything else.
Developer provides a .deb/.rpm?	**Download and install with apt/dnf** — if the developer provides it directly (Chrome, VS Code, Slack), this is fine. It usually sets up a repo for future updates.
Ubuntu only, no alternative?	**Snap** — use it when it's the only option, or when it's the most up-to-date version. Avoid when a Flatpak or native package exists.
Just trying something out?	**AppImage** — download, run, delete. No installation, no cleanup. Good for testing before committing, or for tools you use rarely.

10. Keeping Software Up to Date

Regular updates keep your system secure and your software current. On a desktop system, your distro's update tool usually notifies you when updates are available. You can also update manually:

```
# Ubuntu / Mint — update everything (native packages + any added repos)
$ sudo apt update && sudo apt upgrade -y && sudo apt autoremove -y

# Fedora — update everything
$ sudo dnf upgrade --refresh -y

# Update Flatpak apps (run regardless of distro)
$ flatpak update -y

# A complete update routine (Ubuntu with Flatpak):
$ sudo apt update && sudo apt upgrade -y && flatpak update -y && sudo apt autorem
```

> **How often to update?** Once a week is reasonable for a desktop system. Security updates are important — don't leave a system unpatched for months. Your desktop environment will notify you of available updates; don't dismiss these indefinitely.

Chapter Summary

Method	Command / tool	Best for
Graphical (Ubuntu)	Ubuntu Software / GNOME Software	Browsing and installing without using the terminal
Graphical (Mint)	Software Manager	Best graphical package manager; native .deb packages, ratings, screenshots
apt (terminal)	`sudo apt install <pkg>`	All Debian/Ubuntu-based systems. Fast, precise, handles dependencies
dnf (terminal)	`sudo dnf install <pkg>`	Fedora, AlmaLinux, Rocky Linux, RHEL

Method	Command / tool	Best for
Flatpak	`flatpak install flathub <id>`	Software not in repos; cross-distro; sandboxed; auto-updates
Snap	`sudo snap install <pkg>`	Ubuntu-focused; some software only available as Snap
AppImage	Download → `chmod +x` → run	Portable apps, one-off use, no installation required
.deb / .rpm file	`sudo apt install ./file.deb`	Software distributed directly by developers (Chrome, VS Code)
Full update	`sudo apt update && sudo apt upgrade -y && flatpak update -y`	Keep everything current — run weekly

Next: Chapter 11 — Essential terminal skills. You've been seeing terminal commands throughout this course. Now we explain the terminal properly — navigation, file operations, sudo, and a few key commands that every Linux user needs to know.

Chapter 11 — Essential Terminal Skills

You've seen terminal commands throughout this course — for updating software, changing settings, and fixing problems. Now it's time to understand the terminal properly. This chapter covers exactly what you need to be capable at the command line: reading the prompt, navigating the file system, working with files and folders, using sudo safely, and editing text files. You won't become a shell scripter from this chapter, but you'll be able to follow any guide online and handle most everyday tasks confidently.

> **The terminal is a tool, not a test.** You don't have to memorise everything here — keep this page bookmarked and refer back to it. With a few weeks of regular use the most common commands become second nature. Everything else you look up when you need it, just like any other skill.

1. Opening a Terminal

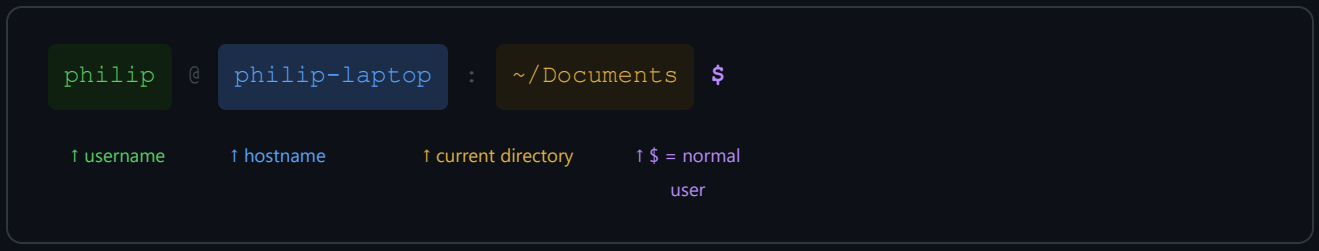
Every Linux desktop has a terminal emulator — an application that gives you a command-line interface. The fastest ways to open one:

- **Ubuntu (GNOME)** — press `Ctrl+Alt+T`, or search for "Terminal" in the Activities overview
- **Linux Mint (Cinnamon)** — press `Ctrl+Alt+T`, or right-click the desktop → "Open Terminal Here"
- **KDE Plasma** — press `Ctrl+Alt+T` (if configured), or search for "Konsole" in the application launcher
- **XFCE** — right-click the desktop → "Open Terminal Here", or find "Terminal Emulator" in the application menu
- **Any desktop** — search "terminal" in the application launcher; every desktop has one installed

> **Terminal emulator vs shell.** The terminal emulator (GNOME Terminal, Konsole, xterm) is the window. The **shell** is the program running inside it that interprets your commands — almost always **Bash** on Linux. When people say "open a terminal" they mean open the terminal emulator; when they say "run this in bash" they mean type it at the prompt in that window.

2. Reading the Prompt

When you open a terminal you see a **prompt** — a line indicating the shell is ready for input. It looks something like this:



- **username** — the account you're logged in as
- **hostname** — the name of the computer (from Chapter 6)
- **current directory** — where in the file system you are right now. The tilde (`~`) is shorthand for your home directory (`/home/philip`)
- **$ symbol** — means you're a normal user. A **# symbol** means you're root (the administrator). You should almost never see # in normal use.

Type a command after the prompt and press **Enter** to run it. The output appears on the lines below, then a new prompt appears when the command finishes.

3. The Linux File System

Linux has one unified file system tree starting at / (called "root" or "slash"). There are no drive letters like Windows — everything, including additional drives and USB sticks, appears as a folder somewhere under /.

```
/ ← the root of the entire file system ├── home/ ← user home directories │ └── philip/ ← your home directory (~) │ ├── Documents/ │ ├── Downloads/ │ ├── Desktop/ │ └── .config/ ← hidden config files (dot files) ├── etc/ ← system configuration files ├── var/ ← variable data: logs, databases, mail │ └── log/ ← system log files ├── usr/ ← installed programs and libraries │ └── bin/ ← most user commands live here ├── bin/ ← essential system commands ├── tmp/ ← temporary files (cleared on reboot) ├── dev/ ← device files (disks, USB, etc.) ├── proc/ ← virtual: running processes └── mnt/ media/ ← mount points for drives/USB sticks
```

The most important location for day-to-day use is your home directory. Everything you create and most software configuration lives under `/home/yourusername/`, abbreviated to `~` in the prompt and in commands.

> **Hidden files** (called "dot files") start with a dot: `.bashrc`, `.config/`, `.ssh/`. They are hidden by default in the file manager and in `ls` output. Press `Ctrl+H` in the file manager to show them, or use `ls -a` in the terminal.

4. Navigating the File System

```
# Print Working Directory — where am I now?
$ pwd
/home/philip

# List files and folders in the current directory
$ ls
Desktop  Documents  Downloads  Music  Pictures  Videos

# List with details (permissions, size, date)
$ ls -l

# List including hidden files (dot files)
$ ls -a

# Both — detailed and including hidden files
$ ls -la

# Change Directory — move into a folder
$ cd Documents
```

```
$ pwd
/home/philip/Documents

# Go back up one level (.. means "parent directory")
$ cd ..

# Go to your home directory (three equivalent ways)
$ cd
$ cd ~
$ cd /home/philip

# Go to an absolute path (starts with /)
$ cd /etc

# Go back to the previous directory (handy for toggling between two locations)
$ cd -
```

> **Tab completion is your best friend.** Start typing a file or folder name and press **Tab** — the shell completes it for you. If there are multiple matches, press Tab twice to see them all. This saves enormous amounts of typing and avoids typos. Use it constantly.

5. Working with Files and Folders

```
# Create a directory
$ mkdir my-project

# Create nested directories in one command (-p = create parents too)
$ mkdir -p projects/website/images

# Create an empty file
$ touch notes.txt

# Copy a file
$ cp notes.txt notes-backup.txt

# Copy a file into a directory
$ cp notes.txt Documents/

# Copy a directory and all its contents (-r = recursive)
```

```
$ cp -r my-project my-project-backup

# Move / rename a file
$ mv notes.txt notes-renamed.txt

# Move a file into a directory
$ mv notes.txt Documents/

# Delete a file
$ rm old-file.txt

# Delete a directory and all its contents (-r = recursive)
$ rm -r old-project/
```

> **rm is permanent.** There is no Recycle Bin, no Trash, no undo. When you `rm` a file it is gone immediately. Be especially careful with `rm -r` (recursive delete) — a typo can delete far more than you intended. Always double-check the path before pressing Enter. Never run `rm -rf /` or `rm -rf ~` — these would destroy your entire system or home directory respectively.

6. Viewing File Contents

```
# Print the entire contents of a file to the screen
$ cat /etc/hostname
philip-laptop

# View a file page by page (press Space to advance, q to quit)
$ less /etc/apt/sources.list

# Show just the first 10 lines of a file
$ head /var/log/syslog

# Show just the last 10 lines of a file
$ tail /var/log/syslog

# Watch a log file update in real time (Ctrl+C to stop)
$ tail -f /var/log/syslog

# Search inside a file for a word or pattern
$ grep "error" /var/log/syslog
```

```
# Search case-insensitively
$ grep -i "wifi" /var/log/syslog
```

> **Use `less`, not `cat`, for large files.** `cat` dumps everything to the screen at once — on a 10,000-line log file you'll see thousands of lines fly past. `less` lets you scroll at your own pace. Inside `less`: `/searchterm` to search, `n` for next match, `q` to quit.

7. sudo — Running Commands as Administrator

Many system tasks — installing software, editing system files, restarting services — require administrator privileges. Linux uses **sudo** (superuser do) to grant temporary elevated permissions for a single command.

```
# Run a single command with admin privileges
$ sudo apt install vlc
[sudo] password for philip:

# sudo caches your password for ~15 minutes — you won't be asked again
# for subsequent sudo commands in that session

# Edit a system file that requires admin access
$ sudo nano /etc/hostname

# Run as root for multiple commands (exit when done — don't stay as root)
$ sudo -i
root@philip-laptop:~#
root@philip-laptop:~# exit
$
```

> **sudo tips and cautions:**
>
> - When typing your sudo password, **nothing appears on screen** — no dots, no asterisks. This is normal; keep typing and press Enter.
> - Only your own account (or other accounts in the `sudo` group) can use sudo — this protects the system from other users.

- Copy-pasting sudo commands from the internet can be dangerous. Read what a command does before running it with sudo.
- If you get "philip is not in the sudoers file", your account needs to be added to the sudo group — ask Chapter 6's user management section.

8. Editing Files with nano

nano is the friendliest terminal text editor — it shows keyboard shortcuts at the bottom of the screen so you don't need to memorise anything to get started. It's available on virtually every Linux system and is what most guides recommend for beginners.

```
# Open or create a file in nano
$ nano myfile.txt

# Edit a system file (requires sudo)
$ sudo nano /etc/hosts
```

Inside nano, the bottom two lines show the available commands. The `^` symbol means the `Ctrl` key:

Ctrl+O
Save (Write Out). Press Enter to confirm the filename.

Ctrl+X
Exit. Asks to save if you have unsaved changes.

Ctrl+W
Search (Where Is). Type your search term and press Enter.

Ctrl+K
Cut the current line (or selected text).

Ctrl+U
Paste (Uncut) the last cut text.

Ctrl+G
Help — full list of shortcuts.

Ctrl+A / Ctrl+E
Jump to start / end of the current line.

Alt+U
Undo last change.

Ctrl+C
Show current line and column number (does NOT copy in nano).

The typical nano workflow: open the file, make your edits, press `Ctrl+O` then `Enter` to save, then `Ctrl+X` to exit. Or just press `Ctrl+X` — if there are unsaved changes it asks "Save modified buffer?" — press `Y` then `Enter`.

9. Terminal Shortcuts That Save Time

Shortcut	What it does
Tab	Auto-complete file/folder names and commands. Press twice to show all matches. Use it constantly.
↑ / ↓	Scroll through command history. Find and re-run a previous command without retyping it.
Ctrl+R	Reverse search through command history. Start typing a word from a previous command to find it.
Ctrl+C	Cancel (stop) the currently running command. Press this when a command hangs or you change your mind.
Ctrl+L	Clear the screen (same as the `clear` command). Keeps your command history.
Ctrl+A	Jump to the beginning of the current command line.
Ctrl+E	Jump to the end of the current command line.
Ctrl+U	Delete everything from the cursor to the beginning of the line. Quick way to clear a command.
Ctrl+W	Delete the word immediately before the cursor.
Ctrl+D	Exit the shell / close the terminal (like typing `exit`).
Ctrl+Shift+C	Copy selected text from the terminal (not Ctrl+C — that cancels the command!).
Ctrl+Shift+V	Paste into the terminal.

10. Pipes and Redirection

Two features of the shell that you'll encounter in guides and find genuinely useful:

The pipe — `|`

A pipe sends the output of one command as the input to another. This lets you chain commands together to filter or process output.

```
# Show running processes and search for a specific one
$ ps aux | grep firefox
```

```
# List files and search the output for .log files
$ ls /var/log | grep "syslog"

# See command history and search it
$ history | grep "apt install"

# Count how many lines of output a command produces
$ ls /usr/bin | wc -l
1847
```

Output redirection — `>` and `>>`

Instead of printing output to the screen, redirect it to a file.

```
# Save command output to a file (overwrites if file exists)
$ ls -la > file-list.txt

# Append output to an existing file (doesn't overwrite)
$ echo "New entry" >> notes.txt

# Discard output entirely (send to /dev/null — the black hole)
$ some-noisy-command > /dev/null 2>&1
```

11. A Few More Useful Commands

SYSTEM INFORMATION

`uname -r`	Show current kernel version
`uptime`	How long the system has been running
`df -h`	Disk usage — how full each partition is
`free -h`	RAM usage — total, used, available
`top`	Live view of running processes and CPU/RAM use. Press q to quit.
`htop`	Nicer version of top (install with apt if missing)

FINDING THINGS

`find ~ -name "*.txt"`	Find all .txt files in your home directory
`grep -r "word" ~/Documents`	Search for "word" inside files recursively
`which firefox`	Find where a command/program is installed
`man ls`	Show the manual page for any command. Press q to quit.

`ls --help`	Quick help for a command (faster than man)

SERVICES AND PROCESSES

`sudo systemctl status nginx`	Check if a service is running
`sudo systemctl restart nginx`	Restart a service
`sudo systemctl enable nginx`	Start a service automatically on boot
`kill 1234`	Stop a process by its PID number
`killall firefox`	Stop all processes named firefox

PERMISSIONS

`chmod +x script.sh`	Make a file executable
`chmod 644 file.txt`	rw-r--r-- (owner write, others read)
`chown philip file.txt`	Change file owner to philip
`ls -l`	Show permissions for files in current directory

> **When a command fails:** read the error message — Linux error messages are usually informative. "Permission denied" means you need sudo. "command not found" means the program isn't installed (try `sudo apt install commandname`). "No such file or directory" means you've typed a path that doesn't exist — check spelling and case (Linux filenames are case-sensitive).

Chapter Summary

Skill	Key commands
Navigate	`pwd` (where am I), `ls` (list files), `cd folder` (enter folder), `cd ..` (go up), `cd ~` (go home)
Files & folders	`mkdir` (create dir), `touch` (create file), `cp` (copy), `mv` (move/rename), `rm` (delete — permanent!)
View files	`cat` (print), `less` (page by page), `head`/`tail` (first/last lines), `grep` (search inside)
Admin	`sudo command` — runs command as administrator. Password cached for ~15 minutes. Nothing shows when typing the password.

Skill	Key commands
Edit files	`nano filename` — Ctrl+O to save, Ctrl+X to exit, Ctrl+W to search.
Shortcuts	Tab (complete), ↑↓ (history), Ctrl+C (cancel), Ctrl+L (clear), Ctrl+R (search history)
Pipes	`cmd1 | cmd2` — send output of cmd1 to cmd2. `> file` — redirect output to file. `>> file` — append to file.
Get help	`man command` — full manual. `command --help` — quick reference. Error messages usually tell you exactly what went wrong.

Next: Chapter 12 — Users, permissions & security. We look at Linux user accounts and groups in more depth, understand file permissions (chmod and chown), sudo configuration, and how to set up the UFW firewall.

Chapter 12 — Users, Permissions & Security

Linux was designed from the start as a multi-user system — multiple people can use the same machine, each with their own files, settings, and level of access. The permission system that makes this work also provides a strong security model: programs and users can only touch what they're explicitly allowed to. This chapter explains how users and groups work, how to read and change file permissions, and how to set up a basic firewall for a desktop machine.

1. Users and the Root Account

Every person who uses a Linux system has a **user account** with a unique username and a numeric **UID** (User ID). The system uses UIDs internally; usernames are just the human-readable label.

There is one special account: **root** (UID 0). Root has unrestricted access to everything on the system — it can read, write, and delete any file, kill any process, and change any setting. On desktop Linux, you are never logged in as root directly; instead you use `sudo` to temporarily borrow root's powers for a single command.

```
# See your own username and UID
$ id
uid=1000(philip) gid=1000(philip) groups=1000(philip),4(adm),27(sudo),1000(philip

# See all user accounts on the system
$ cat /etc/passwd | grep -v nologin | grep -v false
root:x:0:0:root:/root:/bin/bash
philip:x:1000:1000:Philip:/home/philip:/bin/bash

# Check which groups your account belongs to
$ groups
philip adm cdrom sudo dip plugdev lpadmin
```

> **System accounts vs user accounts.** `/etc/passwd` contains many accounts with `/usr/sbin/nologin` as their shell — these are service accounts (www-data for Apache, mysql for MySQL, etc.) that can't log in interactively. They exist so that services run under a restricted identity, not as root. Filtering them out with `grep -v nologin` shows only the real people accounts.

Managing users from the terminal

```
# Create a new user (interactive — sets password and details)
$ sudo adduser sarah

# Delete a user (keeps their home directory)
$ sudo deluser sarah

# Delete a user AND their home directory
$ sudo deluser --remove-home sarah

# Change a user's password
$ sudo passwd sarah

# Change your own password
$ passwd

# Lock an account (prevents login without deleting it)
$ sudo passwd -l sarah

# Unlock it again
$ sudo passwd -u sarah
```

2. Groups

A **group** is a collection of users that share access to certain resources. Every file has both an owner (a user) and an owning group. Groups let you grant access to multiple users at once without making files world-readable.

When you install Linux and create your account, a group with the same name as your username is created automatically (your **primary group**). You also get added to several

system groups that grant specific permissions:

- **sudo** (Ubuntu/Debian) / **wheel** (Fedora/Red Hat) — allows use of `sudo` to run admin commands
- **adm** — read access to system log files in `/var/log`
- **cdrom** — access to optical disc drives
- **plugdev** — access to removable storage devices (USB sticks)
- **lpadmin** — manage printers
- **docker** — run Docker containers without sudo (added when you install Docker)

```
# Add a user to a group
$ sudo usermod -aG groupname username
# The -a means "append" — without it, the user is REMOVED from all other groups!

# Examples:
$ sudo usermod -aG sudo sarah        # Give sarah admin access (Ubuntu)
$ sudo usermod -aG wheel sarah       # Give sarah admin access (Fedora)
$ sudo usermod -aG docker philip     # Let philip run Docker

# Create a new group
$ sudo groupadd developers

# View all groups and their members
$ cat /etc/group | grep developers
```

> **Group membership takes effect at next login.** If you add yourself to a group, you need to log out and back in (or run `newgrp groupname` in the terminal) before the new group permissions apply. Running `groups` after `usermod` won't show the new group until you log out.

3. Understanding File Permissions

Every file and directory in Linux has a set of permissions that controls who can read it, write to it, and execute it. You can see these by running `ls -l`:

```
$ ls -l ~/Documents
-rw-r--r-- 1 philip philip 4096 Jun 15 10:23 report.txt
drwxr-xr-x 2 philip philip 4096 Jun 14 09:00 Projects
-rwxr-xr-x 1 philip philip  512 Jun 13 14:11 backup.sh
```

The first column is the permission string. Let's break it down:

What r, w, and x mean

- **r (read, value 4)** — on a file: can view the contents. On a directory: can list what's inside (`ls`).
- **w (write, value 2)** — on a file: can modify or delete the contents. On a directory: can create, rename, or delete files inside it.
- **x (execute, value 1)** — on a file: can run it as a program or script. On a directory: can enter it (`cd`) and access its contents.
- **- (dash)** — permission is not granted.

The first character — file type

- **-** — regular file
- **d** — directory
- **l** — symbolic link (a shortcut to another file or directory)
- **b / c** — block or character device (disks, terminals)

Numeric (octal) permissions

Permissions are often written as three digits — each digit is the sum of r(4), w(2), x(1) for owner, group, and others respectively:

- **644** — `rw-r--r--` — standard file (owner writes, everyone reads)
- **755** — `rwxr-xr-x` — standard directory or executable (owner full, others read+execute)
- **700** — `rwx------` — private to owner only
- **600** — `rw-------` — private file (SSH keys use this)
- **777** — `rwxrwxrwx` — everyone can do anything (avoid this)

4. chmod — Changing Permissions

chmod (change mode) sets file and directory permissions. You can use numeric notation or symbolic notation.

```
# Numeric notation — set exact permissions
$ chmod 644 report.txt      # rw-r--r-- (standard file)
$ chmod 755 script.sh       # rwxr-xr-x (executable script)
$ chmod 700 private-dir/    # rwx------ (private directory)
$ chmod 600 ~/.ssh/id_rsa   # rw------- (private SSH key)

# Symbolic notation — add or remove specific permissions
$ chmod +x script.sh        # Add execute for everyone
$ chmod -x script.sh        # Remove execute for everyone
$ chmod u+x script.sh       # Add execute for owner only (u=user/owner)
$ chmod g+w shared-file.txt # Add write for group (g=group)
$ chmod o-r secret.txt      # Remove read from others (o=other)
$ chmod a+r readme.txt      # Add read for all (a=all)

# Apply recursively to a directory and all its contents
$ chmod -R 755 my-website/
```

COMMON NUMERIC PERMISSIONS

400	Read-only for owner (e.g., licence files)
600	Private file — owner read/write (SSH keys)

SYMBOLIC NOTATION REFERENCE

u	user (owner)
g	group
o	others

644	Normal file — owner writes, group/others read
700	Private directory or script
755	Standard directory or executable
775	Shared directory — owner and group write

a	all (u + g + o)
+	add permission
-	remove permission
=	set exact permission

5. chown — Changing Ownership

chown (change owner) changes who owns a file or directory. Only root (via sudo) can change file ownership.

```
# Change owner of a file
$ sudo chown philip report.txt

# Change owner AND group together (owner:group)
$ sudo chown philip:developers report.txt

# Change only the group (note the leading colon)
$ sudo chown :developers shared-folder/

# Change ownership recursively
$ sudo chown -R philip:philip ~/Projects/

# A common use case: fixing ownership after copying files as root
$ sudo chown -R $USER:$USER ~/Downloads/extracted-archive/
```

> **When do you need chown?** The most common scenario is when files have been created or copied by root (e.g., via sudo) and the resulting files are owned by root, making them hard to edit as a normal user. `sudo chown -R $USER:$USER path/` transfers ownership back to you. The `$USER` variable automatically expands to your username.

6. sudo — A Bit More Detail

Chapter 11 introduced sudo. Here's a bit more of what it can do and how it's configured.

```
# Run a command as root
$ sudo apt update

# Run a command as a different user (not root)
$ sudo -u www-data ls /var/www/html

# Open an interactive root shell (use sparingly — exit when done)
$ sudo -i

# Re-run the last command with sudo (when you forgot to add it)
$ sudo !!

# Check whether you can use sudo (and what commands)
$ sudo -l

# Clear the sudo password cache (forces re-entry of password)
$ sudo -k
```

The sudoers file

sudo's configuration lives in `/etc/sudoers` and the directory `/etc/sudoers.d/`. This controls who can use sudo and what commands they're allowed to run. **Always edit sudoers with `visudo`** — it validates the syntax before saving, preventing a broken sudoers file that would lock you out.

```
# Edit the sudoers file safely
$ sudo visudo
```

```
# Example sudoers entries:
# Allow user sarah to run any command as root:
sarah ALL=(ALL:ALL) ALL

# Allow sarah to run apt without entering a password:
sarah ALL=(ALL) NOPASSWD: /usr/bin/apt

# The sudo group line (present by default on Ubuntu):
%sudo ALL=(ALL:ALL) ALL
```

> **Never give NOPASSWD: ALL to a regular user account.** This would let anyone who gains access to that account have unrestricted root access without needing a password — completely bypassing the security model. NOPASSWD should be restricted to specific commands only, and only when genuinely necessary.

7. UFW — The Uncomplicated Firewall

Linux uses `iptables` (or the newer `nftables`) for packet filtering, but configuring them directly is complex. **UFW** (Uncomplicated Firewall) is a front-end that makes firewall management straightforward. It's pre-installed on Ubuntu and available on most Debian-based distros.

> **Do you need a firewall on a desktop?** On a home desktop behind a router (which acts as a NAT firewall), the risk is lower — incoming connections from outside your home network are blocked by the router. UFW is still worthwhile as a second layer of defence, and is strongly recommended on any laptop that connects to public WiFi or any machine acting as a server.

Setting up UFW

```
# Check if UFW is installed
$ sudo ufw status
Status: inactive

# Set default policies: block all incoming, allow all outgoing
$ sudo ufw default deny incoming
$ sudo ufw default allow outgoing

# Allow specific services (before enabling — don't lock yourself out)
$ sudo ufw allow ssh       # Allow SSH (port 22) — important if remote

# Enable the firewall
$ sudo ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Common UFW rules

```
# Allow by service name (UFW knows common services)
$ sudo ufw allow ssh        # port 22
$ sudo ufw allow http       # port 80
$ sudo ufw allow https      # port 443

# Allow by port number
$ sudo ufw allow 8080
$ sudo ufw allow 5432       # PostgreSQL

# Allow from a specific IP address only
$ sudo ufw allow from 192.168.1.10

# Allow from a specific IP to a specific port
$ sudo ufw allow from 192.168.1.0/24 to any port 22

# Deny a specific port
$ sudo ufw deny 23          # Block telnet

# Delete a rule
$ sudo ufw delete allow 8080

# Check all current rules
$ sudo ufw status verbose
```

After enabling with the defaults above, here's what the status looks like for a basic desktop that only allows SSH:

```
Status: active

To                         Action                      From
--                         ------                      ----
22/tcp                     ALLOW IN                    Anywhere
22/tcp (v6)                ALLOW IN                    Anywhere (v6)
```

> **Enable UFW before connecting remotely.** If you're configuring a remote machine over SSH, add the SSH allow rule *before* enabling UFW. If you enable UFW without allowing SSH first, you'll be locked out immediately and will need physical access to the machine to recover.

UFW on Fedora

Fedora uses **firewalld** instead of UFW. The graphical tool is **firewall-config**; the command-line tool is **firewall-cmd**. The concepts are the same — default zones, allowing services by name — but the syntax differs:

```
# Fedora — allow a service permanently
$ sudo firewall-cmd --permanent --add-service=http
$ sudo firewall-cmd --reload

# Check current rules
$ sudo firewall-cmd --list-all
```

8. Good Security Habits for Desktop Linux

Keep the system updated

Most real-world attacks exploit known vulnerabilities that have already been patched. Running `sudo apt update && sudo apt upgrade` weekly keeps security patches applied. Don't ignore update notifications.

Use strong, unique passwords

Your login password protects your sudo access. Use a password manager (Bitwarden, KeePassXC) to generate and store strong passwords. Avoid reusing passwords across accounts.

Don't use root for daily work

Never log in as root or stay in a root shell longer than necessary. The permission system only protects you if you're operating as a normal user. A mistake as root can destroy the system; as a normal user, the damage is limited to your own files.

Be careful with sudo commands from the internet

Before running a `sudo` command you found online, understand what it does. `sudo rm -rf /`, `sudo chmod -R 777 /`, and `curl url | sudo bash` are examples that have caused real data loss for people who ran them without checking.

Lock your screen

Set your screen to lock automatically after a few minutes of inactivity (Settings → Privacy → Screen Lock on Ubuntu). Physical access to an unlocked machine bypasses all file permissions.

Enable full-disk encryption

On laptops, LUKS encryption (set up during installation as covered in Chapter 5) protects all your data if the machine is stolen. Without it, anyone with physical access can read your files by booting from a live USB.

Only install software from trusted sources

Stick to your distro's official repositories, Flatpak (Flathub), and well-known developer websites. Random scripts from forums, unverified PPAs, and pirated software are the most common vectors for malware on Linux.

Check file permissions on sensitive files

SSH private keys should be `600` (owner read-only). Configuration files with passwords should not be world-readable. Run `ls -la` in sensitive directories occasionally to check nothing has been accidentally made too permissive.

Chapter Summary

Topic	Key commands and concepts
Users	`sudo adduser name` — create. `sudo deluser name` — delete. `sudo passwd name` — change password. `id` — see your own UID and groups.
Groups	`sudo usermod -aG groupname user` — add user to group (always use `-a`). `groups` — see your current groups. Takes effect at next login.
Permissions	Three sets (owner / group / others) × three bits (r=4, w=2, x=1). `ls -l` shows permissions. Common values: 644 (file), 755 (dir/exec), 600 (private), 700 (private dir).
chmod	`chmod 644 file` (numeric) or `chmod u+x file` (symbolic). `chmod -R` applies recursively to directories.
chown	`sudo chown user:group file` — change owner and group. `sudo chown -R $USER:$USER path/` — reclaim ownership of a directory tree.
sudo	Temporary root access for one command. Configure with `sudo visudo` (never edit sudoers directly). `sudo -l` shows what you're permitted to run.
UFW firewall	`sudo ufw default deny incoming` → `sudo ufw allow ssh` → `sudo ufw enable`. Check rules with `sudo ufw status verbose`. Fedora uses firewalld instead.

Next: Chapter 13 — Keeping Linux healthy. The final chapter covers running updates, monitoring disk space, reading log files, and simple backup strategies to keep your system running reliably long-term.

Chapter 13 — Keeping Linux Healthy

A Linux system that is well maintained stays fast, stable, and secure for years. Neglect it — let updates pile up, disk space fill to the brim, logs balloon unchecked — and small problems compound into big ones. This final chapter covers the four pillars of desktop Linux maintenance: **updates**, **disk space**, **logs**, and **backups**.

1. Keeping the System Updated

Updates deliver security patches, bug fixes, and new features. On a Linux desktop you should run a full update at least once a week — more often if you're running a system exposed to the internet.

Full update routine — Ubuntu / Debian / Mint

```
# 1. Refresh the package index
$ sudo apt update

# 2. Upgrade all installed packages
$ sudo apt upgrade -y

# 3. Upgrade packages that need to remove or add others (new kernel versions etc.
$ sudo apt full-upgrade -y

# 4. Remove packages that are no longer needed
$ sudo apt autoremove -y

# 5. Update Flatpak apps (if you use Flathub)
$ flatpak update -y

# Do it all in one line:
$ sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove -y && flatpa
```

Full update routine — Fedora

```
$ sudo dnf upgrade --refresh -y && flatpak update -y
```

> **apt upgrade vs apt full-upgrade.** `apt upgrade` only upgrades packages that don't require removing or installing additional packages. `apt full-upgrade` (formerly `dist-upgrade`) also handles packages that need dependency changes — necessary for kernel updates. For daily use, `upgrade` is fine. Run `full-upgrade` weekly to catch kernel and library updates.

Checking for and applying kernel updates

```
# See which kernel you're currently running
$ uname -r
6.8.0-45-generic

# List installed kernels
$ dpkg --list | grep linux-image
linux-image-6.8.0-44-generic   ... installed (old — can be removed)
linux-image-6.8.0-45-generic   ... installed (current)

# A reboot is required after a kernel update
$ sudo reboot
```

> **Automatic security updates.** Ubuntu ships with `unattended-upgrades` pre-installed. It automatically applies security-only updates in the background. You can check its configuration at `/etc/apt/apt.conf.d/50unattended-upgrades`. It's a good safety net, but it doesn't replace doing a full upgrade manually — it only handles security patches, not all available updates.

Upgrading to a new distro version

Ubuntu releases a new LTS version every two years. When a new LTS is available, you'll see a notification in the Software Updater. You can also upgrade from the terminal:

```
# Upgrade Ubuntu to the next LTS version
$ sudo do-release-upgrade

# For Fedora (e.g., F39 → F40)
```

```
$ sudo dnf system-upgrade download --releasever=40
$ sudo dnf system-upgrade reboot
```

> **Always back up before a major version upgrade.** Upgrading between distro versions (Ubuntu 22.04 → 24.04, Fedora 39 → 40) is usually smooth, but it is a significant operation that touches almost every package on the system. If something goes wrong, a backup is the only reliable recovery path.

2. Managing Disk Space

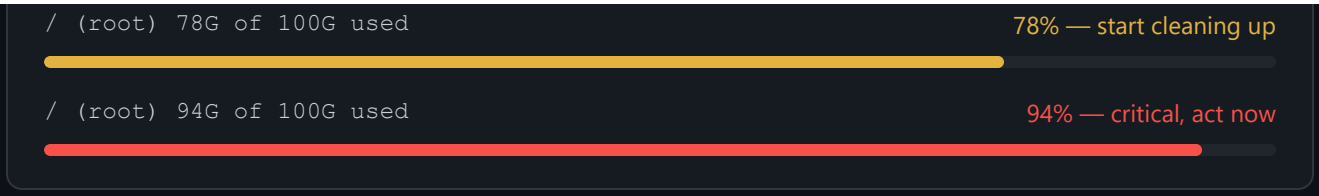
A full disk is one of the most disruptive things that can happen to a Linux system — applications crash, logs stop writing, and in some cases the desktop itself freezes. Check disk usage regularly and clean up before you run out of space.

Checking disk usage

```
# Show disk usage for all mounted filesystems (-h = human-readable)
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda2       100G   42G   55G  44% /
/dev/sda1       512M  6.1M  506M   2% /boot/efi
tmpfs           3.9G  1.2M  3.9G   1% /tmp

# Show disk usage of a directory and all its subdirectories
$ du -sh ~/Downloads
14G     /home/philip/Downloads

# Find the top 10 largest items in a directory
$ du -sh ~/* | sort -rh | head -10

# Find files larger than 500MB anywhere on the system
$ sudo find / -type f -size +500M 2>/dev/null
```

Here's what healthy vs warning disk usage looks like for your root partition (`/`):

```
/ (root) 44G of 100G used                    44% — healthy
```



Freeing up disk space

```
# Remove packages that are no longer needed as dependencies
$ sudo apt autoremove -y

# Clean the apt package cache (downloaded .deb files accumulate here)
$ sudo apt clean            # Remove all cached packages
$ sudo apt autoclean        # Remove only outdated cached packages

# See how much space the apt cache is using
$ du -sh /var/cache/apt/archives/
2.1G    /var/cache/apt/archives/

# Remove old kernels (Ubuntu keeps the last two — autoremove handles this)
$ sudo apt autoremove --purge

# Remove Flatpak runtimes that are no longer used by any app
$ flatpak uninstall --unused -y

# Clear systemd journal logs older than 2 weeks (covered more in the next sectior
$ sudo journalctl --vacuum-time=2weeks

# See the Trash and clear it
$ du -sh ~/.local/share/Trash/
$ rm -rf ~/.local/share/Trash/*
```

> **Don't delete files from /var, /usr, or /lib manually.** These system directories contain files managed by the package manager. Deleting individual files here can break the system in ways that are hard to recover from. Use `apt remove`, `apt purge`, and `apt autoremove` instead — they cleanly handle all the dependencies.

GUI tools for disk space

- **Disk Usage Analyzer** (Baobab) — `sudo apt install baobab` — treemap view of where your space is being used. Excellent for finding unexpected large files visually.
- **BleachBit** — `sudo apt install bleachbit` — cleans browser caches, temp files, and more. Run without root for user files, with root for system files.
- **ncdu** — `sudo apt install ncdu` — terminal-based interactive disk usage browser. Fast and works over SSH.

```
# ncdu — navigate with arrow keys, d to delete, q to quit
$ ncdu ~
```

3. Reading and Managing Log Files

Linux keeps detailed records of what the system and its services are doing. These logs are invaluable when something goes wrong — they tell you exactly what happened and when. Modern Linux uses **systemd journal** for most logging, supplemented by traditional plain-text log files in `/var/log/`.

The systemd journal — journalctl

```
# Show all logs from the current boot
$ journalctl -b

# Follow new log entries in real time (like tail -f for the system)
$ journalctl -f

# Show logs for a specific service
$ journalctl -u NetworkManager
$ journalctl -u apache2
$ journalctl -u ssh

# Show only errors and critical messages
$ journalctl -p err -b

# Show logs since a specific time
$ journalctl --since "2024-06-15 09:00" --until "2024-06-15 10:00"
$ journalctl --since "1 hour ago"
```

```
# Show logs from the previous boot (useful after a crash)
$ journalctl -b -1

# Check how much disk space the journal is using
$ journalctl --disk-usage
Archived and active journals take up 512.0M in the file system.

# Trim journal logs older than 2 weeks / to a maximum size
$ sudo journalctl --vacuum-time=2weeks
$ sudo journalctl --vacuum-size=500M
```

Traditional log files in /var/log/

/var/log/syslog

General system messages. The first place to look when something is misbehaving. (Ubuntu/Debian)

/var/log/auth.log

All authentication events — logins, sudo commands, SSH attempts, failed passwords. Check this if you suspect unauthorised access.

/var/log/kern.log

Kernel messages. Useful for diagnosing hardware problems, driver errors, and filesystem issues.

/var/log/dpkg.log

Record of every package installed, upgraded, or removed by apt/dpkg. Useful for auditing what changed.

/var/log/apt/history.log

Higher-level apt command history — dates, packages, and whether the operation was manual or automatic.

/var/log/Xorg.0.log

X11 display server log. Check here for graphics driver problems, resolution issues, or display not starting.

/var/log/apache2/

Apache web server logs — access.log (every HTTP request) and error.log (problems). Present only if Apache is installed.

/var/log/dmesg

Kernel ring buffer from boot. Same as running `dmesg`. Useful for hardware detection issues at startup.

```
# Read a log file with less (arrow keys, q to quit, / to search)
$ sudo less /var/log/syslog

# Follow a log file in real time
$ sudo tail -f /var/log/syslog

# Search a log for a specific term
$ sudo grep "error" /var/log/syslog
```

```
$ sudo grep "Failed password" /var/log/auth.log

# See recent hardware messages (great for diagnosing USB / disk issues)
$ dmesg | tail -30
$ dmesg | grep -i "error\|fail\|usb"
```

Limiting journal size permanently

By default the journal has no hard size limit. To prevent it growing indefinitely, edit `/etc/systemd/journald.conf` and add or uncomment these lines:

```
SystemMaxUse=500M
SystemKeepFree=200M
MaxRetentionSec=4week
```

```
# Apply the changes
$ sudo systemctl restart systemd-journald
```

4. Quick System Health Checks

A handful of commands give you a snapshot of how the system is doing right now:

```
# How long the system has been running, load averages, number of users
$ uptime
 14:32:01 up 7 days,  3:12,  1 user,  load average: 0.45, 0.38, 0.32

# CPU and memory usage — interactive, press q to quit
$ top
$ htop            # Friendlier version: sudo apt install htop

# Memory usage summary
$ free -h
               total        used        free      shared  buff/cache   available
Mem:           7.7Gi       2.4Gi       3.1Gi       312Mi       2.2Gi       4.9Gi
Swap:          2.0Gi          0B       2.0Gi

# Check if any services have failed
```

```
$ systemctl --failed
  UNIT                          LOAD   ACTIVE SUB    DESCRIPTION
● bluetooth.service             loaded failed failed Bluetooth service

# See recent errors from the system journal
$ journalctl -p err -b --no-pager | tail -20

# Check disk health with SMART (requires smartmontools)
$ sudo apt install smartmontools
$ sudo smartctl -H /dev/sda
SMART overall-health self-assessment test result: PASSED
```

> **Load average explained.** The three numbers after "load average:" in `uptime` are the system load averaged over the last 1, 5, and 15 minutes. A value of 1.0 on a single-core machine means 100% CPU use. On a 4-core machine, values below 4.0 are fine. Values consistently above your core count mean the system is overloaded and processes are queuing for CPU time.

5. Backup Strategies

Linux makes it easy to lose data — `rm` has no Recycle Bin, and a command like `sudo dd` to the wrong device can wipe a disk instantly. A backup is your safety net for both accidents and hardware failure. The golden rule is **3-2-1**: three copies, on two different media types, with one copy offsite (or in the cloud).

Easy

Déjà Dup (GUI)

```
sudo apt install deja-dup
```

Built-in to GNOME as "Backups". Backs up your home directory to a USB drive or cloud storage (Google Drive, Nextcloud). Scheduled automatic backups with encryption. Best option for beginners.

Easy

Timeshift (System snapshots)

```
sudo apt install timeshift
```

Snapshots the system (not personal files) using rsync or BTRFS snapshots. Like Windows System Restore — useful for rolling back after a bad update or config change. Pre-installed on Linux Mint.

Intermediate

rsync (command-line)

```
rsync -avh --delete ~/Documents
/mnt/backup/
```

Intermediate

Restic (cloud + encryption)

```
sudo apt install restic
```

Powerful, flexible tool that synchronises files to another location. Only transfers changed files, making subsequent backups fast. Excellent for scripting and scheduling with cron. Available by default on most distros.

Modern backup tool with deduplication and encryption built in. Backs up to local disk, SSH server, S3, Backblaze B2, and more. Excellent for offsite cloud backups where you don't want the cloud provider to see your files.

A basic rsync backup script

```
# Back up ~/Documents to an external USB drive mounted at /mnt/backup
$ rsync -avh --delete ~/Documents/ /mnt/backup/Documents/

# Flags explained:
#   -a  archive mode (preserves permissions, timestamps, symlinks)
#   -v  verbose output (shows files being transferred)
#   -h  human-readable sizes
#  --delete  removes files from destination that no longer exist in source

# Dry run first — shows what WOULD be done without doing it
$ rsync -avhn --delete ~/Documents/ /mnt/backup/Documents/
```

Scheduling backups with cron

cron is the Linux task scheduler — it runs commands at set times without you having to do anything. Edit your cron schedule with `crontab -e`:

```
# Open your personal crontab for editing
$ crontab -e
```

Cron uses a five-field time format: `minute hour day month weekday`

Cron entry	What it does
`0 2 * * * /path/to/backup.sh`	Run backup script every day at 2:00 AM
`0 3 * * 0 rsync -avh ~/Documents/ /mnt/usb/`	rsync every Sunday at 3:00 AM
`*/30 * * * * /usr/bin/my-check.sh`	Run a script every 30 minutes
`0 9 1 * * /usr/bin/monthly-report.sh`	Run on the 1st of every month at 9:00 AM
`@reboot /home/philip/startup.sh`	Run once at every system startup

> **Crontab quick tip.** Use `crontab -l` to list your current scheduled jobs without editing. If you're not sure your cron syntax is correct, the website **crontab.guru** lets you type a cron expression and see in plain English exactly when it will run.

What to back up

- **Always:** `~/` — your home directory contains all personal files, browser profiles, application config (`~/.config`, `~/.local`), SSH keys (`~/.ssh`), and shell history (`~/.bashrc`)
- **If you have a server:** `/etc/` (all configuration files), `/var/www/` (web content), database dumps (use `mysqldump` for MySQL)
- **Less critical:** `/usr/local/` (manually installed software) — most of this can be reinstalled
- **Not worth backing up:** `/tmp`, `/proc`, `/sys`, `/dev` — these are virtual or temporary and should not be included in backups

6. Monthly Maintenance Checklist

- ☐ **Run a full system update** — `sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove -y && flatpak update -y`
- ☐ **Check disk space** — `df -h`. If root is above 80%, investigate with `ncdu ~` and `sudo apt clean`
- ☐ **Verify backups ran** — check Déjà Dup history, or `ls -lh /mnt/backup/` if using rsync manually. Test that you can restore a file
- ☐ **Review failed services** — `systemctl --failed`. Investigate and restart or remove anything that shouldn't be failing
- ☐ **Check for errors in logs** — `journalctl -p err -b --no-pager | head -30`
- ☐ **Clean the journal** — `sudo journalctl --vacuum-time=4weeks` if you haven't set a permanent limit
- ☐ **Reboot** — especially after kernel updates. Clears memory, applies the new kernel, and reveals any boot issues in a controlled moment rather than unexpectedly

- [ ] **Check Timeshift snapshots** (if using) — ensure a recent snapshot exists before any major update or change
- [ ] **Review UFW rules** — `sudo ufw status verbose`. Remove rules for services you no longer run
- [ ] **Check for suspicious login attempts** — `sudo grep "Failed password" /var/log/auth.log | tail -20`

Chapter Summary

Topic	Key commands and tools
System updates	`sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove -y` weekly. `flatpak update -y` for Flatpak apps. Reboot after kernel updates.
Disk space	`df -h` — overview. `du -sh ~/path/` — size of a directory. `ncdu ~` — interactive browser. `sudo apt clean && sudo apt autoremove` — reclaim space from package cache and old dependencies.
Logs	`journalctl -b` (this boot), `journalctl -f` (live tail), `journalctl -u service` (one service), `journalctl -p err` (errors only). Traditional files in `/var/log/`. Vacuum with `--vacuum-time`.
Health checks	`uptime` (load average), `free -h` (memory), `systemctl --failed` (broken services), `htop` (interactive CPU/memory). `smartctl -H /dev/sda` for disk health.
Backups	Déjà Dup (GUI, home directory). Timeshift (system snapshots, rollback). `rsync -avh --delete src/ dest/` (scriptable, fast). Restic (encrypted, offsite cloud). Schedule with `crontab -e`.

Introduction to Linux — Course Complete

You've covered everything from choosing a distro and installing Linux, through configuring hardware and software, to securing your system and keeping it healthy. Two appendices remain as quick-reference resources to use as you explore Linux further.

Appendix A — Useful software packages

Appendix B — Terminal command cheat sheet

Appendix A — Useful Software Packages

This appendix is a curated reference of software packages organised by category. For each package the install command is given for both **apt** (Ubuntu, Debian, Linux Mint) and **dnf** (Fedora). Where a Flatpak version from Flathub is available and often preferable (more up to date, sandboxed), it is listed as an alternative in **purple**.

● apt install (Ubuntu/Debian/Mint) ● dnf install (Fedora) ● flatpak install flathub (any distro)

> **Prefer native packages for system tools; Flatpak for desktop apps.** Command-line utilities, servers, and libraries are best installed with apt/dnf — they integrate with the system and update with everything else. For desktop apps like browsers, office suites, and media players, the Flatpak version is often newer and better sandboxed.

Internet & Communication

Package	Description	apt	dnf
Firefox GUI	The default browser on most distros. Installed by default on Ubuntu, Mint, Fedora.	`apt install firefox`	`dnf install firefox`
Chromium GUI	The open-source base of Google Chrome. Privacy-focused alternative.	`apt install chromium-browser`	`dnf install chromium`
Thunderbird GUI	Email client from Mozilla. Handles multiple	`apt install thunderbird`	`dnf install thunderbird`

Package	Description	apt	dnf
	accounts, calendars, RSS feeds.		
Discord GUI	Voice, video, and text chat platform. Download .deb from discord.com, or use Flatpak.	`apt install ./discord.deb`	`dnf install ./discord.rp`
Signal GUI	Encrypted messaging desktop client. Flatpak is the easiest install method.	`— (use Flatpak)`	`— (use Flatpak)`
wget CLI	Download files from the terminal. Supports resuming interrupted downloads.	`apt install wget`	`dnf install wget`
curl CLI	Transfer data with URLs. Used for downloading files and testing APIs.	`apt install curl`	`dnf install curl`
nmap CLI	Network scanner. Discover devices and open ports on your local network.	`apt install nmap`	`dnf install nmap`

Office & Productivity

Package	Description	apt	dnf	Flatp
LibreOffice GUI	Full office suite: Writer (Word), Calc (Excel), Impress (PowerPoint), Base (Access). Pre-installed on many distros.	`apt install libreoffice`	`dnf install libreoffice`	org.l
OnlyOffice GUI	Office suite with excellent Microsoft Office compatibility. Good if you share .docx/.xlsx files frequently.	— (use Flatpak)	— (use Flatpak)	org.c
Obsidian GUI	Markdown-based note-taking app with a graph view. Local files, no cloud required.	— (use Flatpak)	— (use Flatpak)	md.ob
Evince GUI	GNOME's document viewer for PDF, PostScript, and more. Lightweight and fast.	`apt install evince`	`dnf install evince`	org.g
Okular GUI	KDE document viewer. Opens PDF, EPUB, CBR, ODT, and many other formats.	`apt install okular`	`dnf install okular`	org.k
Calibre GUI	E-book manager and	`apt install calibre`	`dnf install calibre`	—

Package	Description	apt	dnf	Flatp
	converter. Manages your library and converts between formats (EPUB, MOBI, PDF).			

Media & Entertainment

Package	Description	apt	dnf	Flat
VLC GUI	The universal media player. Plays virtually every video and audio format without extra codecs.	`apt install vlc`	`dnf install vlc`	org.
mpv GUI	Lightweight, keyboard-driven video player. Preferred by power users; excellent hardware decoding.	`apt install mpv`	`dnf install mpv`	io.m
Rhythmbox GUI	GNOME music player and library manager. Pre-installed on Ubuntu.	`apt install rhythmbox`	`dnf install rhythmbox`	org.
Spotify GUI	Spotify desktop client. Flatpak is the recommended	— (use Flatpak)	— (use Flatpak)	com.

Package	Description	apt	dnf	Flat
	install on Linux.			
Handbrake GUI	Video transcoder. Convert videos between formats, compress for web, rip DVDs.	`apt install handbrake`	`dnf install handbrake-gui`	`fr.h`
ffmpeg CLI	Powerful command-line media conversion and processing tool. Used by many other apps internally.	`apt install ffmpeg`	`dnf install ffmpeg`	—
yt-dlp CLI	Download videos and audio from YouTube and hundreds of other sites.	`apt install yt-dlp`	`dnf install yt-dlp`	—
Steam GUI	Gaming platform from Valve. Excellent Linux support via Proton. Flatpak recommended for sandboxing.	`apt install steam`	— (use Flatpak)	`com.`

Graphics & Image Editing

Package	Description	apt	dnf	Fla
GIMP GUI	GNU Image Manipulation Program. Full-featured raster image editor — the Linux equivalent of Photoshop.	`apt install gimp`	`dnf install gimp`	org
Inkscape GUI	Professional vector graphics editor. Works with SVG files. The Linux equivalent of Illustrator.	`apt install inkscape`	`dnf install inkscape`	org
Darktable GUI	RAW photo editing and darkroom software. Comparable to Adobe Lightroom.	`apt install darktable`	`dnf install darktable`	org
Krita GUI	Digital painting and illustration app. Excellent brush engine; popular with artists and animators.	`apt install krita`	`dnf install krita`	org
Shotwell GUI	Simple photo organiser and basic editor. Good for managing a personal photo library.	`apt install shotwell`	`dnf install shotwell`	org
ImageMagick CLI	Command-line image manipulation. Resize,	`apt install imagemagick`	`dnf install ImageMagick`	—

Package	Description	apt	dnf	Fla
	convert, crop, and process images in bulk scripts.			

Development Tools

Package	Description	apt	dnf
VS Code GUI	Microsoft's popular code editor. Download .deb/.rpm from code.visualstudio.com, or use Flatpak.	`apt install ./code.deb`	`dnf in`
build-essential CLI	Meta-package that installs gcc, g++, make, and other essential C/C++ build tools. Install this first when compiling from source.	`apt install build-essential`	`dnf gr`
git CLI	The standard version control system. Required for almost all development work.	`apt install git`	`dnf in`
python3-pip CLI	Python package installer. Install Python libraries with `pip install package-name`.	`apt install python3-pip`	`dnf in`
nodejs / npm CLI	JavaScript runtime and package manager. Use NodeSource repo or nvm for a more up-to-date version.	`apt install nodejs npm`	`dnf in`
Docker CLI	Container platform. Install via Docker's official script or apt repo — not the distro's	`curl -fsSL https://get.docker.com | sh`	`dnf in`

Package	Description	apt	dnf
	version, which is outdated.		
sqlite3 CLI	Lightweight, self-contained SQL database engine. Useful for development and scripting.	`apt install sqlite3`	`dnf in`
meld GUI	Visual diff and merge tool. Compare files, directories, and version-controlled projects side by side.	`apt install meld`	`dnf in`

System & Administration

Package	Description	apt	dnf
htop CLI	Interactive process viewer — a much friendlier version of `top`. Colour-coded, sortable, shows CPU per core.	`apt install htop`	`dnf install htop`
btop CLI	Modern, visually rich resource monitor. Shows CPU, memory, disk I/O, and network in a single terminal view.	`apt install btop`	`dnf install btop`
ncdu CLI	Interactive terminal disk usage analyser. Navigate directories with	`apt install ncdu`	`dnf install ncdu`

Package	Description	apt	dnf
	arrow keys to find space hogs.		
tmux CLI	Terminal multiplexer. Split your terminal into panes, keep sessions running after disconnecting from SSH.	`apt install tmux`	`dnf install tmux`
rsync CLI	Fast, incremental file synchronisation. Used for backups and copying large directories efficiently. (Usually pre-installed.)	`apt install rsync`	`dnf install rsync`
ufw CLI	Uncomplicated Firewall — easy front-end for iptables. Covered in Chapter 12. Pre-installed on Ubuntu.	`apt install ufw`	`— (use firewalld)`
GParted GUI	Graphical partition editor. Resize, create, delete, and format disk partitions. Essential for disk management.	`apt install gparted`	`dnf install gparted`

Package	Description	apt	dnf
Timeshift GUI	System snapshot and restore tool. Like Windows System Restore. Pre-installed on Mint. Covered in Chapter 13.	`apt install timeshift`	— (use Snapper)
BleachBit GUI	System cleaner. Removes browser caches, temp files, old logs. Frees disk space safely.	`apt install bleachbit`	`dnf install bleachbit`
smartmontools CLI	Read SMART health data from hard drives and SSDs. Detect failing drives before they die.	`apt install smartmontools`	`dnf install smartmontoo`

Terminal Extras

Package	Description	apt	dnf	Flatpak alterna
zsh CLI	Alternative shell to bash. Powerful tab completion, history, and theme support. Pair with Oh My Zsh for plugins.	`apt install zsh`	`dnf install zsh`	—
bat CLI	`cat` with syntax highlighting, line numbers, and git diff integration.	`apt install bat`	`dnf install bat`	—

Package	Description	apt	dnf	Flatpak alterna
	Drop-in replacement for `cat`.			
fzf CLI	Fuzzy file finder for the terminal. Press Ctrl+R for fuzzy history search; integrates with vim and tmux.	`apt install fzf`	`dnf install fzf`	–
ripgrep CLI	Extremely fast recursive text search. Respects .gitignore. Much faster than grep for large codebases.	`apt install ripgrep`	`dnf install ripgrep`	–
tree CLI	Display directory contents as an indented tree. `tree -L 2` shows two levels deep.	`apt install tree`	`dnf install tree`	–
jq CLI	Command-line JSON processor. Parse, filter, and format JSON output from APIs and config files.	`apt install jq`	`dnf install jq`	–
neofetch CLI	Display system information (distro, kernel, DE, RAM, CPU) alongside an	`apt install neofetch`	`dnf install neofetch`	–

Package	Description	apt	dnf	Flatpak alterna
	ASCII art logo. A classic.			
tldr CLI	Simplified man pages with practical examples. `tldr tar` shows the 5 most useful tar commands instead of the full manual.	`apt install tldr`	`dnf install tldr`	—

Multimedia Codecs

Out of the box, Ubuntu and Mint can't play MP3s, H.264 video, or DVDs due to patent restrictions. Install the restricted extras package to add these codecs:

```
# Ubuntu — installs MP3, H.264, AAC, and other codecs + Microsoft fonts
$ sudo apt install ubuntu-restricted-extras

# Linux Mint — same effect
$ sudo apt install mint-meta-codecs

# Fedora — enable RPM Fusion first, then install codecs
$ sudo dnf install https://mirrors.rpmfusion.org/free/fedora/rpmfusion-free-relea
$ sudo dnf install https://mirrors.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree
$ sudo dnf install gstreamer1-plugins-{bad-\*,good-\*,base} gstreamer1-plugin-ope

# DVD playback (libdvdcss — Debian/Ubuntu only, legal in most countries)
$ sudo apt install libdvd-pkg && sudo dpkg-reconfigure libdvd-pkg
```

> **Why aren't codecs included by default?** Some codecs (MP3, H.264, AAC) are covered by software patents in certain countries. Ubuntu and Mint ship without them to avoid legal issues, but

> provide an easy way to install them. Fedora's RPM Fusion repository handles the same problem. Once installed, media playback "just works."

Virtual Machines & Remote Access

Package	Description	apt	dnf
VirtualBox GUI	Run Windows or other Linux distros in a virtual machine. Download from virtualbox.org for the latest version.	`apt install virtualbox`	`dnf install virtualbox`
GNOME Boxes GUI	Simple VM manager using KVM/QEMU. Easier than VirtualBox for running Linux VMs. Native to GNOME.	`apt install gnome-boxes`	`dnf install gnome-boxes`
openssh-server CLI	Run an SSH server on your machine so you can log in remotely from other computers.	`apt install openssh-server`	`dnf install openssh-serv`

Package	Description	apt	dnf
Remmina GUI	Remote desktop client. Connect to Windows machines via RDP, or other Linux machines via VNC or SSH.	`apt install remmina`	`dnf install remmina`

Quick Install Blocks

Copy-paste these blocks to set up a fully-equipped system in one go.

Ubuntu / Mint — essential desktop setup

```
$ sudo apt update && sudo apt install -y \
  vlc gimp inkscape libreoffice \
  htop btop ncdu tmux tree bat fzf ripgrep jq tldr neofetch \
  git build-essential curl wget rsync \
  python3-pip sqlite3 \
  ubuntu-restricted-extras
```

Fedora — essential desktop setup

```
# Enable RPM Fusion first
$ sudo dnf install -y \
  https://mirrors.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fedo
  https://mirrors.rpmfusion.org/nonfree/fedora/rpmfusion-nonfree-release-$(rpm -E

$ sudo dnf install -y \
  vlc gimp inkscape libreoffice \
  htop btop ncdu tmux tree bat fzf ripgrep jq tldr neofetch \
  git curl wget rsync \
```

```
  python3-pip sqlite \
  gstreamer1-plugins-good gstreamer1-plugins-bad-free lame
```

Flatpak essentials (any distro)

```
# Add Flathub if not already set up
$ flatpak remote-add --if-not-exists flathub https://dl.flathub.org/repo/flathub.

# Install popular apps
$ flatpak install -y flathub \
  org.mozilla.firefox \
  org.mozilla.Thunderbird \
  org.libreoffice.LibreOffice \
  org.videolan.VLC \
  org.gimp.GIMP \
  com.spotify.Client \
  com.valvesoftware.Steam \
  md.obsidian.Obsidian
```

Next: Appendix B — Terminal command cheat sheet. A quick-reference card for the most useful Linux terminal commands, organised by task.

Appendix B — Terminal Command Cheat Sheet

A quick-reference card for the most commonly used Linux terminal commands, organised by task. Each entry shows the base command, a practical example, and a short note on what it does or what to watch out for. For more detail on any command, run `man command` or `tldr command`.

> **Reading the examples.** Text in `amber` shows a real example with actual arguments filled in. Words in *grey italic* in the notes column are placeholders — replace them with your own file names, paths, or values.

Navigation

Command	Example	What it does
`pwd`	`pwd`	Print working directory — shows where you are right now.
`ls`	`ls -la`	List directory contents. `-l` long format, `-a` show hidden files (those starting with .).
`cd`	`cd ~/Documents`	Change directory. `cd` alone goes home. `cd ..` goes up one level. `cd -` goes back to the previous directory.
`tree`	`tree -L 2`	Display directory as an indented tree. `-L 2` limits depth to 2 levels. Install: `apt install tree`.

Files & Directories

Command	Example	What it does
`touch`	`touch notes.txt`	Create an empty file, or update the timestamp of an existing one.
`mkdir`	`mkdir -p projects/web`	Create a directory. `-p` creates parent directories too — no error if they already exist.

Command	Example	What it does
`cp`	`cp -r src/ backup/`	Copy files or directories. `-r` required for directories (recursive).
`mv`	`mv old.txt new.txt`	Move or rename a file or directory. Works across directories too.
`rm`	`rm -r old-folder/`	Delete files or directories. **Permanent — no Recycle Bin.** `-r` for directories, `-i` to confirm each deletion.
`ln -s`	`ln -s /etc/nginx nginx`	Create a symbolic link (shortcut). The link points to the target; deleting the link does not delete the target.
`find`	`find ~ -name "*.log"`	Search for files by name, type, size, or date. `-type f` files only, `-type d` directories only, `-size +100M` larger than 100 MB.
`file`	`file mystery.bin`	Identify the type of a file by its contents, not just its extension.
`du`	`du -sh ~/Downloads`	Disk usage of a directory. `-s` summary total, `-h` human-readable sizes.

Viewing File Contents

Command	Example	What it does
`cat`	`cat config.txt`	Print an entire file to the terminal. Good for short files. Use `less` for long files.
`less`	`less /var/log/syslog`	Page through a file. Space = next page, **b** = back, **/** = search, **q** = quit.
`head`	`head -20 file.txt`	Show the first N lines of a file (default 10).
`tail`	`tail -f /var/log/syslog`	Show the last N lines. `-f` follows the file in real time — great for watching logs.
`grep`	`grep -i "error" syslog`	Search for a pattern in a file. `-i` case-insensitive, `-r` recursive through directories, `-n` show line numbers, `-v` invert (show non-matching lines).
`wc`	`wc -l file.txt`	Word count. `-l` count lines, `-w` words, `-c` bytes.
`diff`	`diff file1.txt file2.txt`	Show differences between two files line by line.

Editing Files

NANO — BEGINNER-FRIENDLY EDITOR

`nano file.txt` Open or create a file

`Ctrl+O` Save (Write Out) — press Enter to confirm

`Ctrl+X` Exit (prompts to save if unsaved)

`Ctrl+W` Search for text

`Ctrl+K` Cut current line

`Ctrl+U` Paste cut text

`Alt+U` Undo

`Ctrl+C` Show cursor position (NOT copy!)

VIM — POWERFUL MODAL EDITOR

`vim file.txt` Open file (starts in Normal mode)

`i` Enter Insert mode (to type)

`Esc` Return to Normal mode

`:w` Save the file

`:q` Quit (fails if unsaved changes)

`:wq` Save and quit

`:q!` Force quit without saving

`/word` Search (n = next match)

Pipes & Redirection

`|`	Pipe — send output of one command as input to another: `ps aux | grep firefox`
`>`	Redirect output to a file (overwrites): `ls > filelist.txt`
`>>`	Redirect output and append (does not overwrite): `echo "line" >> file.txt`
`<`	Read input from a file: `sort < names.txt`
`2>`	Redirect stderr (error output) to a file: `cmd 2> errors.log`
`2>&1`	Redirect stderr to the same place as stdout: `cmd > out.log 2>&1`
`&`	Run command in the background: `long-task.sh &`
`;`	Run commands in sequence regardless of success: `cmd1 ; cmd2`
`&&`	Run second command only if the first succeeded: `apt update && apt upgrade`
`||`	Run second command only if the first failed: `mkdir dir || echo "exists"`

Permissions & Ownership

Command	Example	What it does
`ls -l`	`ls -l ~/Documents`	Show permissions, owner, group, size, and date for each file.

Command	Example	What it does
`chmod`	`chmod 755 script.sh`	Change permissions. Numeric: 4=read, 2=write, 1=execute. Three digits for owner/group/others. `-R` recursive.
`chmod +x`	`chmod +x script.sh`	Add execute permission for all users (symbolic notation).
`chown`	`sudo chown philip:philip file`	Change owner and group. `-R` recursive. Requires sudo.
`id`	`id`	Show your UID, GID, and all group memberships.
`groups`	`groups`	List all groups the current user belongs to.
`sudo`	`sudo apt update`	Run a command as root. Password cached for ~15 minutes. Use `sudo -k` to clear the cache.

Common numeric permission values:

600
`rw-------`
Private file — owner reads/writes only (SSH keys)

644
`rw-r--r--`
Normal file — owner writes, everyone reads

700
`rwx------`
Private directory or executable

755
`rwxr-xr-x`
Standard directory or executable script

775
`rwxrwxr-x`
Shared — owner and group write

777
`rwxrwxrwx`
Everyone can do everything — avoid

Package Management

APT — UBUNTU / DEBIAN / MINT

`sudo apt update` Refresh package index

`sudo apt upgrade -y` Upgrade all packages

`sudo apt full-upgrade -y` Upgrade including kernel changes

`sudo apt install pkg` Install a package

DNF — FEDORA / RED HAT

`sudo dnf upgrade --refresh` Update and refresh package index

`sudo dnf install pkg` Install a package

`sudo dnf remove pkg` Remove a package

`sudo dnf autoremove` Remove unused dependencies

`sudo apt remove pkg` Remove package (keep config)

`sudo apt purge pkg` Remove package and config files

`sudo apt autoremove` Remove unused dependencies

`sudo apt clean` Clear download cache

`apt search term` Search for packages

`apt show pkg` Show package details

`dpkg -l | grep pkg` Check if a package is installed

`dnf search term` Search for packages

`dnf info pkg` Show package details

`dnf list installed` List all installed packages

FLATPAK — ANY DISTRO

`flatpak install flathub id` Install an app from Flathub

`flatpak update -y` Update all Flatpak apps

`flatpak list` List installed Flatpak apps

`flatpak uninstall --unused` Remove unused runtimes

Processes & System

Command	Example	What it does
`ps aux`	`ps aux | grep firefox`	List all running processes. Pipe to grep to filter by name.
`top`	`top`	Live process monitor. `q` to quit, `k` to kill a process by PID.
`htop`	`htop`	Friendlier process monitor with colour. F9 to kill, F6 to sort, `q` to quit.
`kill`	`kill 1234`	Send a signal to a process by PID. `kill -9 PID` force-kills it immediately.
`killall`	`killall firefox`	Kill all processes with a given name.
`uptime`	`uptime`	Show how long the system has been running and the CPU load averages (1, 5, 15 min).
`free -h`	`free -h`	Show RAM and swap usage in human-readable format.
`df -h`	`df -h`	Show disk space usage for all mounted filesystems.
`uname -r`	`uname -r`	Show the running kernel version. `uname -a` shows all system info.
`lscpu`	`lscpu`	Show CPU architecture, cores, threads, and cache information.
`lsblk`	`lsblk`	List all block devices (disks and partitions) in a tree view.

Command	Example	What it does
`lsusb`	`lsusb`	List all connected USB devices.
`dmesg`	`dmesg | tail -30`	Kernel ring buffer — hardware events, driver messages, boot messages.
`reboot`	`sudo reboot`	Restart the system. `sudo shutdown -h now` to power off immediately.

Services — systemctl

Command	Example	What it does
`systemctl status`	`systemctl status ssh`	Show the current state of a service — running, stopped, or failed. Shows recent log output too.
`systemctl start`	`sudo systemctl start apache2`	Start a service right now (does not persist after reboot).
`systemctl stop`	`sudo systemctl stop apache2`	Stop a service.
`systemctl restart`	`sudo systemctl restart nginx`	Stop and start a service — applies config changes.
`systemctl reload`	`sudo systemctl reload nginx`	Reload config without stopping the service (not supported by all services).
`systemctl enable`	`sudo systemctl enable ssh`	Enable a service to start automatically at boot.
`systemctl disable`	`sudo systemctl disable bluetooth`	Prevent a service from starting at boot.
`systemctl --failed`	`systemctl --failed`	Show all services that have failed. Check these regularly as part of maintenance.

Networking

Command	Example	What it does
`ip addr`	`ip addr show`	Show all network interfaces and their IP addresses.

Command	Example	What it does
`ip route`	`ip route show`	Show the routing table. The "default via" line shows your gateway.
`hostname -I`	`hostname -I`	Quick way to see your local IP address(es).
`ping`	`ping -c 4 8.8.8.8`	Test connectivity to a host. `-c 4` sends 4 packets then stops.
`traceroute`	`traceroute google.com`	Show the network hops between you and a destination. Install: `apt install traceroute`.
`dig`	`dig google.com`	DNS lookup — resolve a domain name to its IP address.
`nslookup`	`nslookup google.com`	Alternative DNS lookup tool, simpler output than dig.
`ss`	`ss -tuln`	Show open ports and connections. `-t` TCP, `-u` UDP, `-l` listening only, `-n` numeric ports.
`nmcli`	`nmcli device wifi list`	NetworkManager CLI — list WiFi networks, connect, disconnect, manage connections.
`curl`	`curl -I https://example.com`	Fetch a URL. `-I` headers only, `-o file` save to file, `-L` follow redirects.
`wget`	`wget https://example.com/file.zip`	Download a file. Supports resuming with `-c`.
`ssh`	`ssh philip@192.168.1.10`	Connect to a remote machine securely. Add `-p 2222` for a non-standard port.
`scp`	`scp file.txt user@host:~/`	Securely copy a file to/from a remote machine over SSH.

Logs

Command	Example	What it does
`journalctl -b`	`journalctl -b`	All logs from this boot. `-b -1` for the previous boot (useful after a crash).
`journalctl -f`	`journalctl -f`	Follow new log entries in real time.
`journalctl -u`	`journalctl -u ssh`	Logs for a specific service only.
`journalctl -p err`	`journalctl -p err -b`	Show only errors and above from the current boot.

Command	Example	What it does
`journalctl --since`	`journalctl --since "1 hour ago"`	Show logs from a specific time window.
`tail -f`	`tail -f /var/log/syslog`	Follow a traditional log file in real time.

Archiving & Compression

Command	Example	What it does
`tar -czf`	`tar -czf archive.tar.gz folder/`	Create a compressed tar archive (.tar.gz). `c`=create, `z`=gzip, `f`=filename.
`tar -xzf`	`tar -xzf archive.tar.gz`	Extract a .tar.gz archive. Add `-C /path/` to extract to a specific directory.
`tar -tf`	`tar -tf archive.tar.gz`	List contents of a tar archive without extracting.
`zip`	`zip -r archive.zip folder/`	Create a .zip file. `-r` includes subdirectories recursively.
`unzip`	`unzip archive.zip -d /output/`	Extract a .zip file. `-d` specifies the destination directory.
`gzip / gunzip`	`gzip file.txt`	Compress or decompress a single file. Creates `file.txt.gz`; the original is replaced.

Terminal Keyboard Shortcuts

PRODUCTIVITY

- **Tab** — Autocomplete command or file name. Press twice to show all options.
- **↑** / **↓** — Navigate command history.
- **Ctrl** + **R** — Reverse search through history. Keep pressing to go further back.
- **Ctrl** + **L** — Clear the screen (same as the `clear` command).
- **Ctrl** + **A** — Jump to the start of the current line.
- **Ctrl** + **E** — Jump to the end of the current line.

CONTROL

- **Ctrl** + **C** — Cancel / interrupt the running command. NOT copy (use Ctrl+Shift+C).
- **Ctrl** + **Z** — Suspend the running process (send to background). Resume with `fg`.
- **Ctrl** + **D** — Send EOF / exit the current shell or Python/Node REPL.
- **Ctrl** + **U** — Delete everything from cursor to the start of the line.

Alt + ← / →	Jump word by word along the current line.
Ctrl + W	Delete the word immediately before the cursor.
Ctrl + Shift + C	Copy selected text in the terminal.
Ctrl + Shift + V	Paste into the terminal.

Getting Help

Command	Example	What it does
`man`	`man ls`	Full manual page for a command. Navigate with arrow keys, search with `/`, quit with `q`.
`--help`	`ls --help`	Short built-in help summary — usually faster than the full man page for a quick flag reminder.
`tldr`	`tldr tar`	Community-written examples for the most common uses of a command. Install: `apt install tldr`.
`which`	`which python3`	Show the full path of a command — confirms it's installed and which version will run.
`type`	`type ls`	Show whether a command is a binary, alias, shell built-in, or function.
`history`	`history | grep apt`	Show your command history. Pipe to grep to find a specific past command.

Tip: use the man page search. If you're not sure which command you need, `man -k keyword` (or `apropos keyword`) searches all man page summaries for that word. For example, `man -k compress` lists every command related to compression. It won't always be obvious, but it's a useful last resort before reaching for a search engine.

Introduction to Linux — All Chapters & Appendices Complete

You have now covered everything in the course — from choosing a distro and installing Linux to configuring hardware, managing software, securing your system, keeping it healthy, and working confidently in the terminal.

Keep this appendix handy as a day-to-day reference as you explore Linux further. The more you use the terminal, the more natural it becomes.

Ch 1 — What is Linux?
Ch 2 — Choosing a Distro
Ch 3 — Preparing to Install
Ch 4 — Installation Media
Ch 5 — The Install Process
Ch 6 — First Boot Setup
Ch 7 — Desktop Environments
Ch 8 — Sound
Ch 9 — Networking
Ch 10 — Installing Software
Ch 11 — Terminal Skills
Ch 12 — Users & Permissions
Ch 13 — Keeping Linux Healthy
App A — Software Packages
App B — Command Cheat Sheet