

# Getting Started with Claude Code

This guide is free. Share it with anyone.

There is no technical barrier to using AI well. The people charging thousands for prompt engineering workshops are selling you confidence you do not need. Communicating well with AI is a real skill, but it is one you develop through practice, not one you buy in a course. The only thing you need is a guide through the first 30 minutes. This is it.

Claude Code is a tool that lets you have a conversation with AI inside a project folder on your computer. Unlike a standard AI chatbot, it can read your files, edit your documents, build things for you, and remember your preferences across sessions. You can use it for anything from reformatting your CV to building a personal website to summarising a stack of research papers. It stores your instructions in a file that it reads before every conversation, so you never have to re-explain who you are or how you work.

Everyone should be able to use these tools. Not to automate their thinking, but to contribute to the conversation about what AI should and should not be used for. You cannot critically engage with something you have never used. This guide gets you started.

## What you need

A Mac or Windows machine, a Claude Pro subscription (\$20/month), and about 30 minutes. You do not need to know how to code. This guide is useful whether you want to explore AI for research, teaching, administration, governance, or personal projects.

## Step 1: Open your command line

You will type commands into a text-based app on your computer. It looks like a plain window with a blinking cursor. You type a command, press Enter, and the computer does something. Think of it as a chat window for your computer. It feels unfamiliar at first. Within an hour, it will feel normal.

**On Mac:** Press **Cmd + Space**, type **Terminal**, and press Enter. **On Windows:** Click the Start menu and search for **PowerShell**. Everything in this guide happens inside this app. If you have a Claude desktop app, leave it closed. This is different.

## Step 2: Install Claude Code

**Windows users: before installing**, you will need **Git for Windows** (free from git-scm.com). Download and install it first. On university or managed machines, your IT department may need to allow script execution in PowerShell. If the install command below fails, try the WinGet method instead, or ask IT for help.

**Mac** (paste into Terminal):

```
curl -fsSL https://claude.ai/install.sh | bash
```

**Windows** (paste into PowerShell):

```
irm https://claude.ai/install.ps1 | iex
```

**Windows alternative:** `winget install Anthropic.ClaudeCode` (this avoids script execution restrictions on managed machines). Installation may take a few minutes. Your computer may also prompt you to install additional software (on Mac, this is usually Xcode Command Line Tools, which includes Git; this is standard Apple software and is free and safe to install). Say yes. This can take up to 30 minutes. You will see text scrolling in the window. That is normal. Let it run and it will stop on its own. After the main installation, your screen may show one final line to paste. It will look something like: `echo 'export PATH=...'`. If it appears, copy that line, paste it, and press Enter. If nothing appears, skip this step. Once installation finishes, **close your command line app and reopen it**. You only need to install once.

## Step 3: Sign in

Make sure you have a Claude Pro subscription active at [claude.ai](https://claude.ai) before this step. Type `claude` and press Enter. Claude Code will open a browser window where you log in with your Claude account. Once authorised, select the recommended settings when asked. You are now inside Claude Code. Type in your own words, just as you would in any AI chatbot.

## Step 4: Create your project folder

A folder on your computer is called a 'directory' in the command line. Same thing, different word. Type these three lines, pressing Enter after each:

```
mkdir ~/my-project
cd ~/my-project
claude
```

This creates a folder called `my-project` in your home folder, moves into it, and starts Claude Code. Call it whatever makes sense for your work (use hyphens instead of spaces, e.g. `my-research-project`). On Mac, find it in Finder with **Cmd + Shift + H**. On Windows, open File Explorer and go to **C:\Users\[your username]**.

If you have run these commands before and get an error saying the folder already exists, that is fine. Just type the second and third lines to move into the existing folder and start Claude.

### Step 5: Let Claude get to know you

Claude Code can store permanent instructions in a file called `CLAUDE.md` that it reads at the start of every conversation. You do not need to write this yourself. Ask Claude to interview you:

*Interview me: ask me one question at a time about who I am, what I work on, how I write, and what rules you should follow. After the interview, create a `CLAUDE.md` file with everything you have learned. Start now.*

Claude will ask about your name, your role, your field, your writing preferences, your pet hates. Answer naturally. When it has enough, it creates the file. Every future conversation in this folder starts from that file. To change anything later, just tell Claude: 'Add a rule that says never use American spelling.' You never need to open the file manually.

### Step 6: Add your files

Copy files into the `my-project` folder using Finder or File Explorer. Drafts, reading notes, reference material, style guides, previous work you want Claude to learn your voice from. Claude can see everything in the folder. Or just ask Claude: 'Create a subfolder called notes and another called drafts.' It will do it.

**Important:** On Mac, dragging a file into a folder **moves** it. The file disappears from its original location. If you want to keep the original, hold the **Option** key while dragging to make a copy instead. On Windows, dragging between drives copies by default, but dragging within the same drive moves the file.

### Your first small project

The best way to learn is to ask Claude to build something small that you will actually use. Not by reading documentation, but by working through a real task and picking things up as you go.

- **Format your CV.** Drop your existing CV into the folder and ask Claude to redesign it. Tell it what style you want, what to emphasise, what to cut.
- **Build a personal website.** Ask Claude: 'Build me a one-page website with my name, bio, and links to my work.' A full personal website can take as little as two hours.
- **Reformat a document.** Drop a style guide into the folder and ask Claude to reformat an existing document to match. Once it understands the style, it applies it to everything.
- **Summarise a batch of papers.** Drop PDFs into a subfolder and ask Claude to produce a one-page summary of the key arguments and where they agree or disagree.
- **Create a template.** A daily journal, a meeting agenda, a lesson plan. Claude saves it as a file you can reuse and refine.
- **Plan your content.** Ask Claude: 'Create a content calendar for my social media based on these three themes.' Give it your themes and it builds a schedule.

What matters is that you learn how Claude Code works by doing something real. You will make mistakes. Claude will ask for clarification. You will correct it. Each correction teaches both of you: Claude learns your preferences (because you update the `CLAUDE.md`), and you learn what Claude can do.

### How to talk to Claude Code

You do not need to learn any special commands or syntax. Just talk in your own words, exactly as you would talk to any AI chatbot. The difference is that Claude Code can see your files, create new ones, edit existing ones, and remember your preferences across sessions.

Some examples of things you can just say:

- 'Read my draft and tell me where the argument is weakest.'
- 'Rewrite the introduction in a more conversational tone.'
- 'Check all the references in this document and flag any that look wrong.'
- 'Create a folder structure for my new research project.'
- 'Check whether this contradicts anything else I have written in this folder.'
- 'How does this work?' (about anything: a feature, a file, a concept)
-

If you get stuck, ask Claude. If you do not understand something it did, ask it to explain. If it does something you do not like, tell it and ask it to add a rule to your **CLAUDE.md** so it does not happen again. There is no wrong way to use it.

## Voice notes and dictation

You do not have to type everything. Dictation software lets you speak to Claude Code instead. Talk through your ideas out loud, let the software transcribe and Claude formats the result. This is particularly powerful for first drafts: speak your rough thoughts, then ask Claude to structure and polish them in your writing style (which it already knows from your **CLAUDE.md**).

**Wispr Flow** is an excellent dictation app for Mac and Windows that works everywhere, including Terminal and PowerShell. It runs quietly in the background and transcribes as you speak. This referral link gives you a free month to try it: [wisprflow.ai/r?SAM2113](https://wisprflow.ai/r?SAM2113)

## Real examples: what you can build

To give you a sense of what is possible: I am a professor with no software engineering background. Using Claude Code and my own words, I have built:

- **A personal website** built from scratch in two hours. HTML, CSS, responsive design. No coding knowledge required.
- **A professional CV** in a custom style, regenerated from a single command whenever it needs updating.
- **Interactive educational games** played by thousands of people. One asks players to distinguish human writing from AI writing. Another tests whether citations are real or fabricated.
- **A peer review simulator.** Drop in a paper draft, type one command, and two simulated reviewers produce structured feedback.
- **An analytics dashboard** that pulls data, generates daily insights, and flags trends automatically.
- **Beautifully formatted PDFs** (like this one) generated from a single conversation.

None of this required learning to code. It required learning to describe what you want clearly and to refine the instructions over time.

## Security and guardrails

Claude Code can read and edit files on your computer. That is its power, but it also means you should think about what you put in your project folder and what rules you set.

- **Keep sensitive files out of your project folder.** Bank details, passwords, API keys, tax documents. If it is in the folder, Claude can see it. Keep personal and financial files elsewhere.
- **Set boundaries in your CLAUDE.md.** You can add rules like: 'Never share or include the contents of files marked confidential.' 'Always ask before deleting files.' Claude will follow these instructions.
- **Review before you approve.** Claude Code asks permission before taking actions like running commands or editing files. Read what it is proposing before you say yes. This is your checkpoint. Use it.
- **Your guardrails should reflect your values.** If you want Claude to always flag when it is uncertain, add that as a rule. If you want it to never fabricate references, make that explicit. The **CLAUDE.md** is not just a style guide. It is a set of operating principles.
- **If you would not show it to a stranger, do not put it in the folder.** That is a reasonable mental model for what Claude can access.

**A good rule of thumb:** Claude Code's interface runs on your machine, but your prompts and files are sent to Anthropic's servers for processing. As of March 2026, conversations on Claude Pro are not used to train models (verify current policy at [anthropic.com/privacy](https://anthropic.com/privacy)). Do not put student names, patient records, or any personally identifiable information into your project folder. If you are using Claude Code with work materials, check your institution's AI use policy first.

## Power features (when you are ready)

You do not need any of this to get started. But as you get more confident, Claude Code can do more.

**Slash commands** save workflows you use repeatedly. Ask Claude: 'Create a skill called peer-review that simulates two reviewers reading a paper and giving structured feedback.' From then on, type `/peer-review` and it runs the full workflow. You can create skills for anything: summarising papers, writing abstracts, checking references, formatting for a journal.

**Hooks** are things that happen automatically. Ask Claude: 'Every time I start a new session, remind me of any deadlines.' Claude sets it up. It happens every time you open Claude Code in that folder.

You can ask Claude to teach you about these features whenever you are ready. There is no rush.

## Troubleshooting

'**command not found**': Close Terminal (or PowerShell) and reopen it. Check you are typing in your command line app, not in the Claude desktop app or a browser.

**Claude opens in a browser or desktop app:** You want the command line version. Close the desktop app, go back to Terminal or PowerShell, and type `claude` there.

**Payment or upgrade screen:** Normal the first time. Complete sign-up in the browser, then return to your command line.

'**File exists**' error on `mkdir`: The folder already exists from a previous attempt. That is fine. Skip to `cd ~/my-project` and then `claude`.

**Do I need to install again each time?** No. To start a new session: open Terminal or PowerShell, type `cd ~/my-project`, then `claude`. That is it.

## Using AI well: critical AI literacy

Setting up Claude Code is the easy part. The harder question is: now that you have it, how do you use it well?

Most AI guides focus on productivity. Get more done. Work faster. Automate everything. This is not that guide. The goal is not to make you faster. The goal is to make you more deliberate.

**Critical AI literacy** means knowing when to use AI and when to leave it alone. It means understanding what AI is good at (drafting, reformatting, summarising, identifying structural patterns in text) and what it is bad at (original thought, emotional intelligence, ethical judgement, knowing when it is wrong). It means treating AI output as a starting point that requires your expertise, not a finished product that replaces it.

Some principles worth building into your practice:

- **Always verify.** Claude will sound confident even when it is wrong. Check facts, references, and claims against original sources. Do not trust AI output on anything where accuracy matters unless you have confirmed it yourself.
- **Your name goes on it.** If Claude writes a draft for you, that draft becomes your responsibility the moment you submit, publish, or send it. Read everything. Edit everything. If you would not put your name on it without Claude, do not put your name on it with Claude.
- **Notice what you stop doing.** If you find yourself no longer reading closely, no longer drafting from scratch, no longer making judgement calls, ask whether that is a trade worth making. The thinking is the valuable part. AI should support your thinking, not replace it.
- **Be honest about your use.** If a colleague, a journal, an employer, or an institution asks whether you used AI, tell the truth. Disclosure builds trust. Concealment destroys it.
- **Slow down.** The temptation with a powerful tool is to go faster. Resist it. Read the output slowly. Ask yourself what it missed. Ask yourself what you would have written differently. The friction is where the learning happens.

AI is changing how some knowledge work gets done, particularly first-draft writing, document processing, and literature review. The question is whether we direct that change or let it happen to us. Critical AI literacy is the difference. It does not mean rejecting AI. It means engaging with it on your terms, with your values, and with your judgement intact.

## What you are actually building

A standard AI chatbot is a disposable conversation. You type, it responds, the context evaporates. Claude Code is a workspace. Your instructions, your files, your rules are all present every time you sit down. The AI starts from understanding rather than ignorance because it reads your instructions before every conversation.

You do not need to learn Terminal commands or Markdown or any technical syntax. You talk to Claude in your own words, and it builds the infrastructure around you. Every correction you make ('do not use that word', 'always format references like this', 'my supervisor prefers shorter paragraphs') becomes a permanent instruction. Your workspace grows as you work.

The more precisely your instructions describe you, the less time you spend correcting the output. That is the trade worth making.

**Want to go further?** AI AutoBase is a platform and learning hub for people who want to build real AI automation systems, not just learn tools. Get practical workflows, step-by-step training, and a community focused on real results.  
Join now: <https://aiautobase.com/>