Claude Code, from *zero*.

Claude Code turns plain English into working software, in your own project, on your own computer. **You do not need to know how to code.** You need to know what you want and how to check that you got it. The rest of it is on this card.

01 · THE FIRST FIVE MINUTES

`npm install -g @anthropic-ai/claude-code`
Once, ever. Needs Node 18+ from nodejs.org

`cd my-project`
It only ever sees the folder you start it in

`claude`
That is the whole thing. Sign in when the browser opens

`/init`
It reads your project and writes itself a briefing. Never skip

Before you start: Claude Code needs a paid Claude account. Pro is $20 a month and includes it. The free tier does not.

02 · YOUR FIRST TWENTY MINUTES

- [ ] Install Node, then Claude Code
- [ ] Open a terminal in your project folder
- [ ] Type claude and sign in
- [ ] Run /init, then read what it wrote about you
- [ ] Ask it to explain your own project back to you
- [ ] Press Esc mid-answer, so you know that you can

Ask it about your own code, not "hello", so you can judge the answer.

03 · THE LOOP THAT ACTUALLY WORKS

Almost every bad result comes from jumping straight to Build. This is the whole method.

1 Explore
Have it read the code and explain it back, before anything else

2 Plan
Shift+Tab twice. It thinks, it cannot touch a file. Read the plan

3 Build
One outcome per request, so you can tell which one broke

4 Prove
Never accept "done". Ask for the real output, every time

5 Save
Say "commit this" whenever it works. That is your undo button

THEN AROUND AGAIN, ONE SMALL PIECE AT A TIME

04 · SAY IT LIKE THIS

INSTEAD OF
~~"make the site look better"~~

SAY
"On the homepage at 390px wide the hero button wraps onto three lines. Keep it on one line down to 320px, without dropping the text below 16px."

INSTEAD OF
~~"is this any good?"~~

SAY
"Find the three things most likely to break this in production, worst first, with the file and line. Do not tell me what is good about it."

KEEP GOING

The whole guide, free.

Seventeen prompts you can copy with one tap, the CLAUDE.md template, all twelve rules, and a plain English glossary. No signup.

modernmustardseed.com/fieldguide

OR HAND IT OFF

Stuck? Or want it built?

Call **(406) 312-1223** and Mr. Mustard, our own AI agent, picks up at any hour. Or book thirty minutes and bring nothing but the idea.

modernmustardseed.com/book

The reference

Anything in [BRACKETS] is yours to replace. That is the only editing these need.

PROMPTS THAT DO THE WORK

Explain my project to me

YOUR VERY FIRST REAL PROMPT IN ANY CODEBASE

```
Read this project and explain it like I have never seen it: what it does, the five files that matter most, and the one thing most likely to break. No jargon, and do not write any code.
```

Plan it before you build it

EVERY FEATURE BIGGER THAN A TYPO

```
I want to add [THE FEATURE]. Read the code that already does something similar, then plan it: the approach, every file you would change, what could go wrong. Plan only, no code.
```

The error prompt

ANYTHING RED IN YOUR TERMINAL OR BROWSER CONSOLE

```
[PASTE THE WHOLE ERROR] Reproduce this first. Tell me the cause in one sentence. Then fix it and show me the run that proves it.
```

The honest review

BEFORE YOU SHOW ANYONE ANYTHING

```
Review this as the engineer who maintains it after I quit. Every problem, worst first, with the file and line. Do not list strengths.
```

Put it on the internet

YOU HAVE SOMETHING WORKING LOCALLY AND NO IDEA WHAT HAPPENS NEXT

```
I want this live on the internet at a real address. One step at a time, wait for me after each, and tell me any cost before I commit.
```

COMMANDS WORTH KNOWING

`/clear`	Next task, clean slate. Use it more than feels necessary
`/compact`	Long job, same thread, running out of room
`/rewind`	It broke something. Put the code and the chat back
`/init`	First time in a project. Writes the briefing
`/permissions`	Stop approving the same safe command forever
`/agents`	A specialist with its own instructions and memory
`/help`	The true, current list for your version
`claude -c`	Continue the last conversation where you left it
`claude -r`	Pick an older conversation from a list
`claude update`	Get the newest version

WHEN IT GOES SIDEWAYS

It changed files I never mentioned	Rewind, then name the file and the single outcome you want
It got slow and started making dumb mistakes	Clear the session and restate the goal in five lines
It is confidently wrong about a tool or library	Make it read the actual installed source, or run the tool with --help
It says it fixed it and it is not fixed	Ask for the real output, every time, before you believe anything
It broke something that used to work	Rewind, or restore from git, then commit far more often

THE KEYS, AND THE FOUR CHARACTERS THAT DO WORK

`Esc`	`Esc Esc`	`Shift+Tab`	`Up arrow`	`Ctrl+C`
Stop it now, keep everything so far	Go back and edit an earlier message	Cycle: ask, auto-accept, plan mode	Your previous prompts	Cancel. Twice quits

`/` Opens the command list for your exact version	`@` Point at a file or folder as you type	`#` Start a line with this to save a rule into memory permanently	`!` Run a terminal command yourself without leaving the session

THE TWELVE RULES, LEARNED THE EXPENSIVE WAY

01 **Done is not evidence.** Load the real URL yourself. A green terminal is a claim.

02 **Never let it guess at a command.** Make it run `--help`. Invented options look exactly like real ones.

03 **Two failed attempts means the plan is wrong.** Stop patching. Back to plan mode, ask for another approach.

04 **Save at every working state.** "Commit this." Turns any disaster into a five second undo.

05 **Give it the error, all of it.** Paste every line. Your paraphrase deletes the part that solves it.

06 **One session, one job.** Two sessions on the same files overwrite each other quietly.

07 **Secrets never enter the chat.** No keys in prompts, no keys in CLAUDE.md. Not even to test.

08 **Read the plan like it costs money.** Thirty seconds there is the best time you spend all day.

09 **It agrees with you by default.** Never ask "is this right". Ask "what breaks first, and why".

10 **Fresh memory beats clever wording.** Going dumb? Clear it and restate the goal in five lines.

11 **You own the output.** Fastest hands you have ever had. Still your name on it.

12 **Build small, ship it, then grow it.** The only wrong move is planning for a month.