Claude Code

About me

Daniel Schwartz

A UCT Graduate in Computer Science and Psychology

Over 8 years of working in Technology industry

Currently a DevSecOps Engineer working at JUMO



Scan to ask questions



What's in this talk

At a high level

01

Introduction to Claude Code

What it is, and how to get started using Claude Code

02

Claude Code customization

A very hackable and customizable general purpose tool

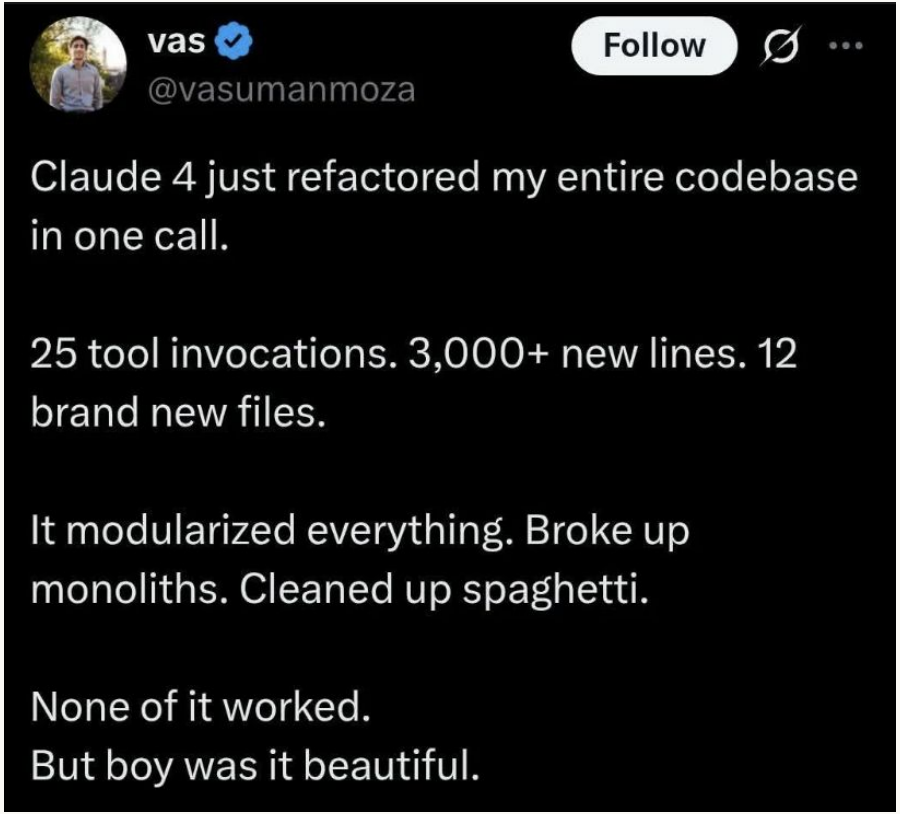
03

How Claude Code has changed how I work

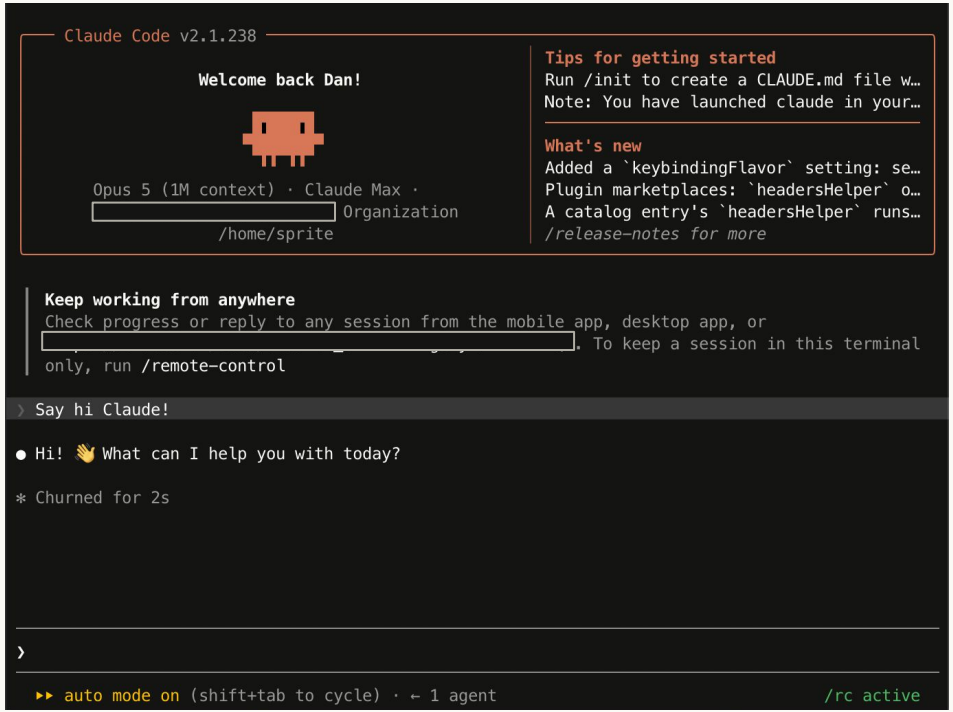
My day-to-day Claude workflow and advice from experience

Welcome to the Rollercoaster

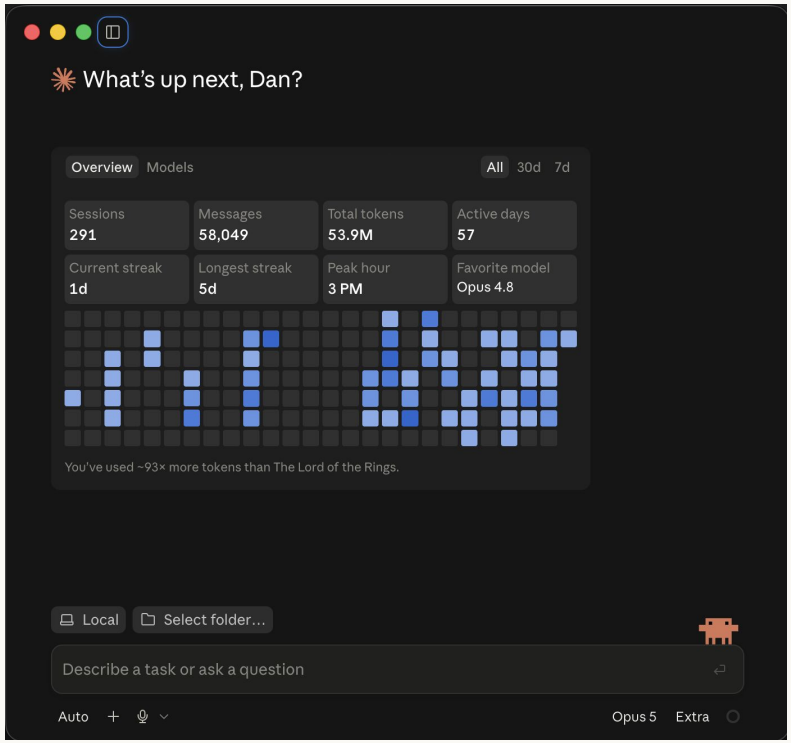
1. Oh wow, it can cook!
2. Oh no, I burnt my food...
3. Wait it's 2am!?
4. What's an IDE again?

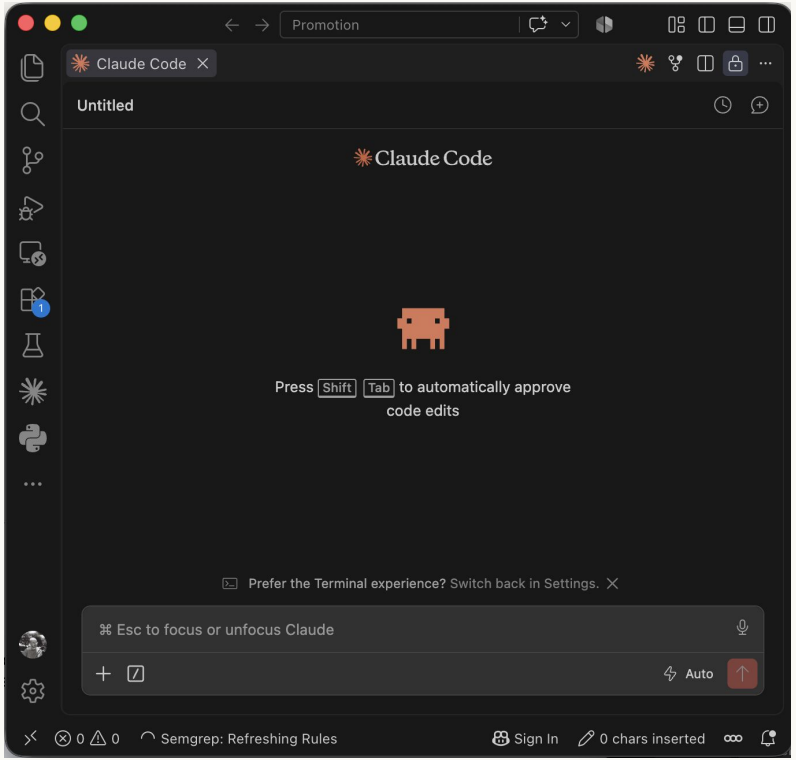


What even is Claude Code?

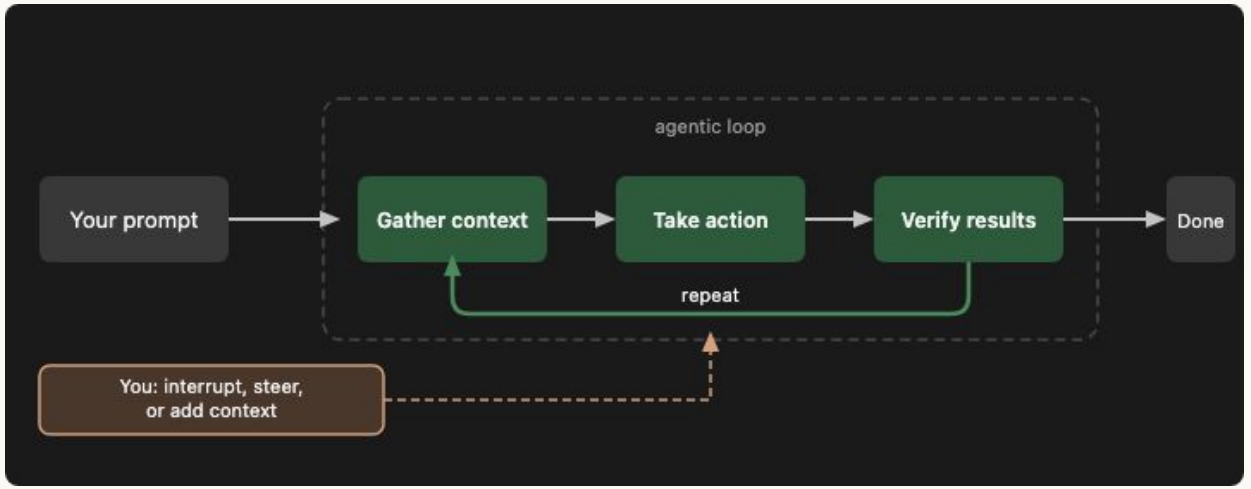


What even is Claude Code?





What even is Claude Code?



https://code.claude.com/docs/en/how-claude-code-works

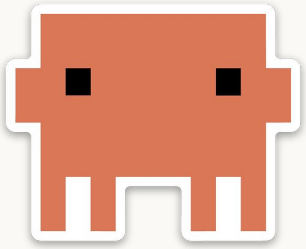
Getting started with Claude Code

macOS/Linux/WSL

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

Windows Powershell

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



Migrate to native installer if not already:

```
$ claude install
```

https://code.claude.com/docs/en/quickstart

The things you manage

Prompts

- Your messages to Claude
- Instructions CLAUDE.md
  - Organization
  - User
  - Project (and sub dirs)
- Memories for that project
- Skills (SKILL.md) that are loaded

Tools

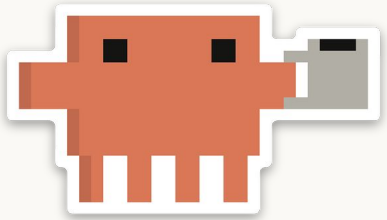
- Your shell, your CLIs, your computer, your access
- MCPs/Connectors you connect
- Your Claude settings and permissions

Context

- Everything Claude can see (Files, Tool results, Instructions, Memories, Images)
- 1 Million tokens context window
- Affects cost and quality of output

Basic commands to get you started

- /init Initialize a new CLAUDE.md for your codebase
- /config Open settings menu
- /powerup Discover Claude Code features with interactive lesson
- /usage Track your session usage, cost, activity
- /doctor Built-in Skill for diagnosing setup issues

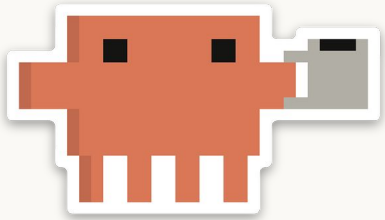


https://code.claude.com/docs/en/commands

Basic commands to get you started

- /usage





Different Modes

shift-tab to cycle







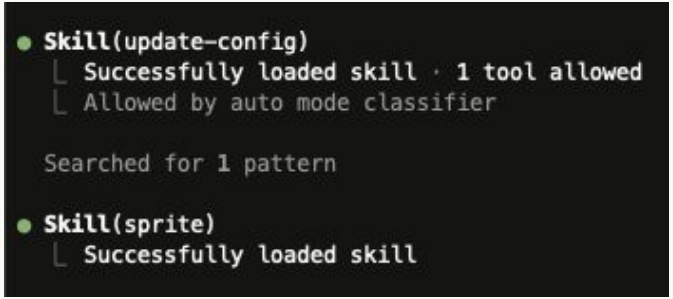
Auto-mode: The new default, excellent for general use and safety

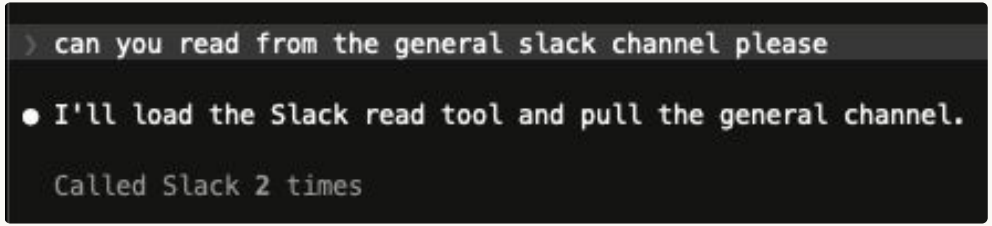
Plan mode: Purely exploring, research and planning a task

Bypass permissions: It is called YOLO mode for a reason

Examples of Components

These add context and tools





Skills - lazy loaded instructions and context to teach Claude how to do a repeated task

E.g. Using a skill to update my Claude config

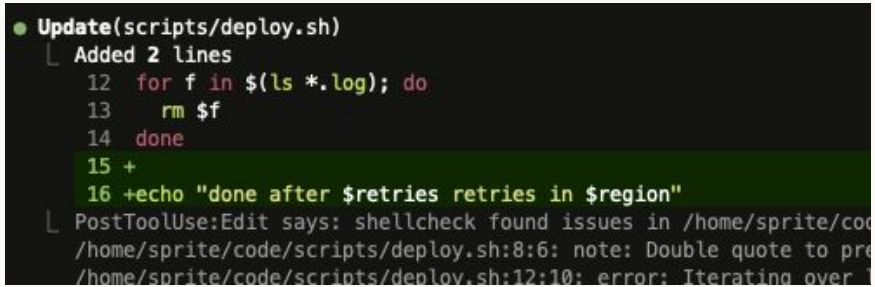
MCPs - tools Claude can call to fetch context from external systems or to make changes to them

E.g. using the Slack connector to read from a channel

https://code.claude.com/docs/en/skills
https://code.claude.com/docs/en/mcp-quickstart

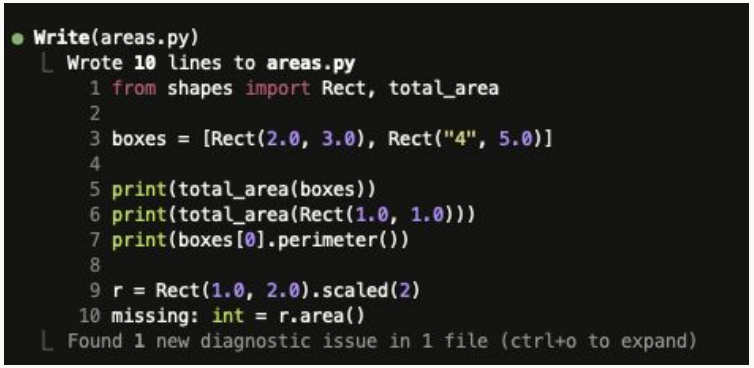
Examples of Components

These add determinism



Hooks - deterministically hook into events to take action at different stages of the workflow

E.g. a PostToolUse hook to run Shellcheck on shell scripts



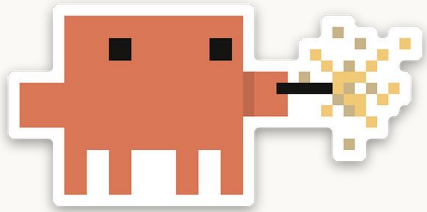
LSPs - Code Intelligence using Language Server Protocols like your IDE does for you

E.g. Python LSP showing issues in the file on Write

https://code.claude.com/docs/en/hooks-guide
https://code.claude.com/docs/en/discover-plugins#code-intelligence

Plugins and Marketplaces

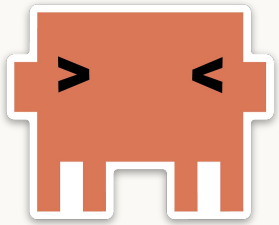
- A plugin is the distributable bundle of all the components that Claude can use (Skills, MCPs, Hooks, Agents, LSPs, Monitors, Themes, Output Styles)
- You can manage your plugins via the /plugin menu
- You install plugins from a marketplace, such as a GitHub repository (*anthropics/claude-plugins-official*)
- You can install any number of marketplaces to source plugins (/plugin marketplace add anthropics/claude-code)
- You can create your own plugins and marketplaces for them to share with others, or reference external plugins in your marketplace



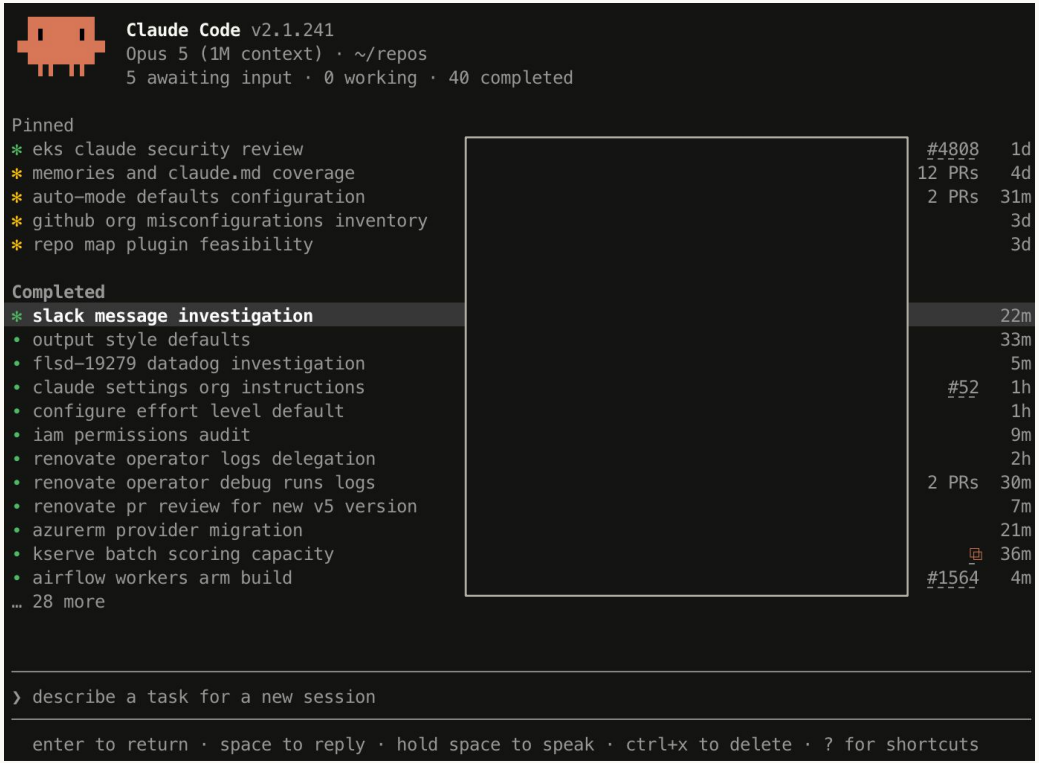
https://code.claude.com/docs/en/discover-plugins

A few nice features to know

- Advisor (/advisor)
- Enable Remote Control (/rc)
- Use Monitors for continuous background polling
- Setup built in Sandbox for Bash (/sandbox)
- Customize Auto-mode Classifier (/setup-auto-mode)
- Set your Output Styles (new “Concise” style)
- Voice mode (/voice), and 20 Languages
- Screen Reader Support, and Colorblind themes (/theme)



My Claude Code workflows



Agent Fleet view (press ←)

My Claude Code workflows

Planning tasks

- Open Claude in parent folder to all my repos
- Start with a conversation
- Definition of Done
- Commit idea to a ticket
- Get agreement with Team

Tackling tasks

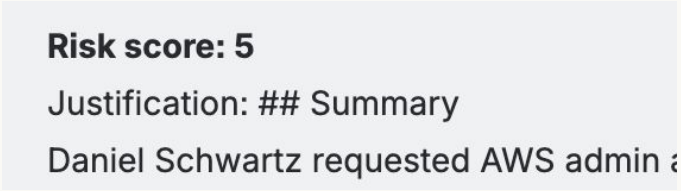
- New session, ticket link
- Sizing up your approach
  - Which model? Effort?
  - Dynamic Workflows?
  - Agent Teams?
- Auto-mode with worktree
- Steer the session
- Finish with draft PRs

Code review

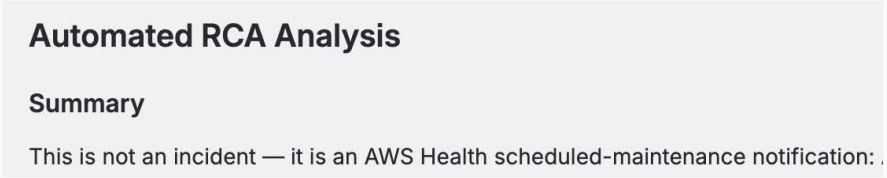
- Human in the loop PR reviews
- Adversarial /code-review
- /security-review or Claude Security plugin
- Ensure documentation is always updated

Agents we created at work that help me sleep at night

1. Access Request Reviewer



2. Automated Root Cause Analysis



Bringing Claude Code to your teams

Education

- Build your own intuition and share it
- Facilitate discussions around automating processes
- Have the hard conversations

Collaboration

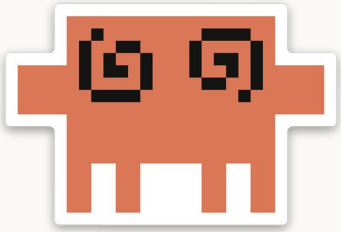
- Create a plugin marketplace for sharing customizations
- Create a feedback channel for working on points of friction
- Work in the open

Safeguards

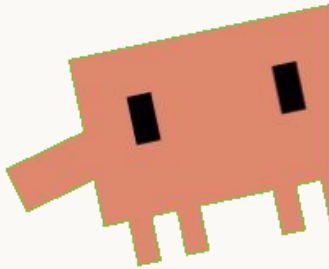
- Setup your managed settings from experience over time
- Standardize your SDLC and tooling
- Lean on auto-mode classifier and sandboxing

Advice from experience

- Managing costs of a session
  - New task, new session. Compact early or not at all.
  - @ relevant files instead of searching
  - Delegate noisy tasks to subagents
- Keep your feedback loop… looping
  - Always define what done looks like
  - Self-check verification loops
  - Lean on your best practice tools
- Getting good results
  - Function of the quality of your plan
  - Keep your context light of bloat
  - Use durable instructions
  - Evaluate your skills
- Seek and give honest feedback
  - Don’t let slop take over!



And most importantly have some fun!



Daniel Schwartz

Slides @justanotherspy/talks on GitHub

Reach out on Linkedin!

Check out /radio!