Claude Code

5 Claude Code skills to install first

Every install command, step by step. Repos and numbers were checked on GitHub on October 16, 2026.

- Superdesign
- Superpowers
- Trail of Bits
- Karpathy skill
- Playwright

Install pack
October 2026

Before you start

1 **Claude Code**, installed and signed in.

Lines in a dark box marked **Type in Claude Code** go into Claude Code's prompt. Claude can't type its own slash commands, so you do.

2 **A terminal** for the lines marked **Run in your terminal**.

Superdesign and Playwright install with npm (Node.js). The Trail of Bits scanners install with Homebrew or pip.

3 **Install only from these five repos.**

github.com/superdesigndev/superdesign-skill · github.com/obra/superpowers · github.com/trailofbits/skills · github.com/forrestchang/andrej-karpathy-skills · github.com/microsoft/playwright-cli

What this guide is not. We didn't build these skills. Licenses: Superdesign skill MIT, Superpowers MIT, Trail of Bits CC-BY-SA-4.0, Karpathy skill MIT (per its README), Playwright CLI Apache-2.0. Superdesign drives the superdesign.dev service, so it needs an account there.

01 / 05 **Superdesign**

Stop shipping AI-slop UI

`github.com/superdesigndev/superdesign-skill · MIT`

It gives Claude an **infinite design canvas**. It branches **a few directions at once** and designs inside **your own design system** (.superdesign/design-system.md).

Superdesign: UI, presentations, and graphics for coding agents

Stop shipping AI-slop UI. Coding agents write great code and mediocre interfaces: generic layouts, default shadcn everything, no taste. Superdesign is the skill that gives your agent design judgment, so the UI it ships actually looks considered.

Install it once and your agent (Claude Code, Cursor, Codex, and 70+ others) can find real design direction and generate + iterate high-quality UI, presentations, and graphics on an infinite canvas, all without leaving your terminal.

> Powered by superdesign.dev, the AI product design agent.

TYPE IN CLAUDE CODE

```
> /plugin marketplace add superdesigndev/superdesign-skill
> /plugin install superdesign@superdesign
```

RUN IN YOUR TERMINAL

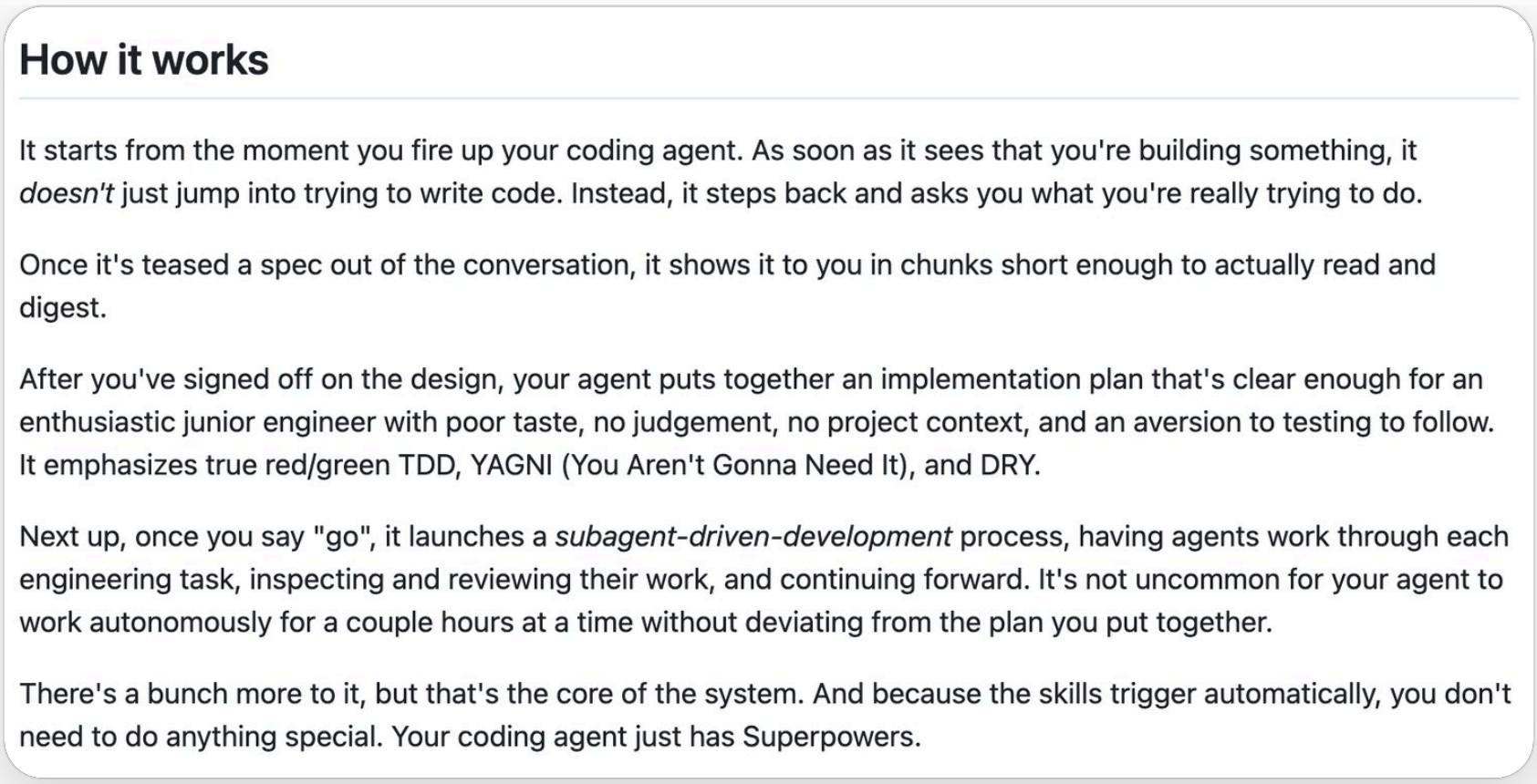
```
$ npm install -g @superdesign/cli@latest
$ superdesign login
```

02 / 05 **Superpowers**

Asks first, then builds with subagents

`github.com/obra/superpowers · MIT`

Before any code, Claude **asks what you're really building** and writes the spec. Then **a fresh subagent builds each task**, with a review after each, and its test-driven skill **deletes code written before its test**.



How it works

It starts from the moment you fire up your coding agent. As soon as it sees that you're building something, it *doesn't* just jump into trying to write code. Instead, it steps back and asks you what you're really trying to do.

Once it's teased a spec out of the conversation, it shows it to you in chunks short enough to actually read and digest.

After you've signed off on the design, your agent puts together an implementation plan that's clear enough for an enthusiastic junior engineer with poor taste, no judgement, no project context, and an aversion to testing to follow. It emphasizes true red/green TDD, YAGNI (You Aren't Gonna Need It), and DRY.

Next up, once you say "go", it launches a *subagent-driven-development* process, having agents work through each engineering task, inspecting and reviewing their work, and continuing forward. It's not uncommon for your agent to work autonomously for a couple hours at a time without deviating from the plan you put together.

There's a bunch more to it, but that's the core of the system. And because the skills trigger automatically, you don't need to do anything special. Your coding agent just has Superpowers.

TYPE IN CLAUDE CODE

```
> /plugin install superpowers@claude-plugins-offic
ial
```

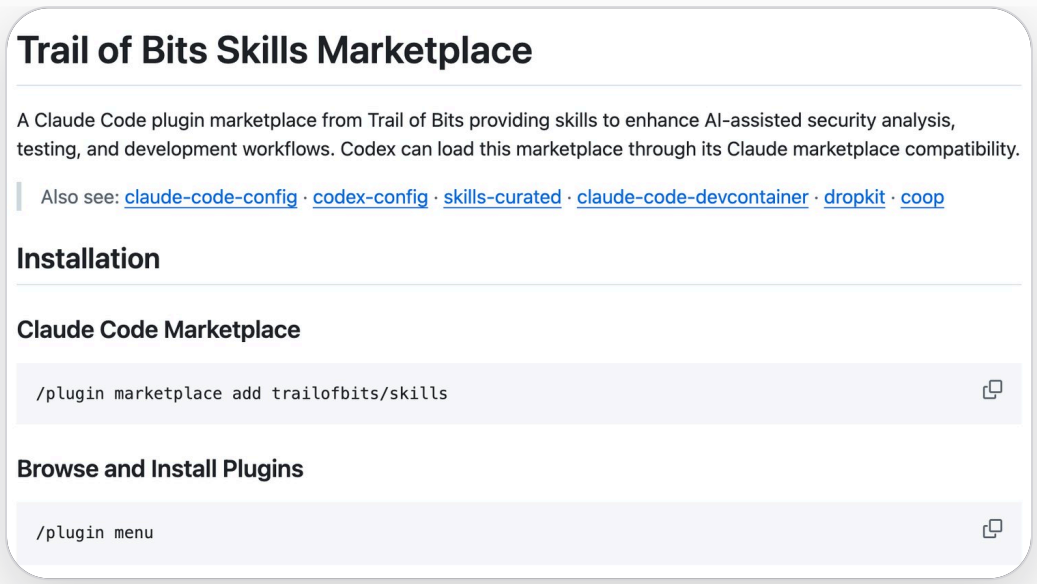
Nothing to call. It starts on its own when you begin building something.

03 / 05 **Trail of Bits**

Real bugs, not a vague warning

`github.com/trailofbits/skills · CC-BY-SA-4.0`

static-analysis runs **CodeQL and Semgrep**, built on Trail of Bits' own Testing Handbook. **fp-check** then gives each suspected bug a **TRUE or FALSE POSITIVE verdict** with evidence.



Trail of Bits Skills Marketplace

A Claude Code plugin marketplace from Trail of Bits providing skills to enhance AI-assisted security analysis, testing, and development workflows. Codex can load this marketplace through its Claude marketplace compatibility.

Also see: claude-code-config · codex-config · skills-curated · claude-code-devcontainer · dropkit · coop

Installation

Claude Code Marketplace

```
/plugin marketplace add trailofbits/skills
```

Browse and Install Plugins

```
/plugin menu
```

TYPE IN CLAUDE CODE

```
> /plugin marketplace add trailofbits/skills
> /plugin install static-analysis@trailofbits
> /plugin install fp-check@trailofbits
```

RUN IN YOUR TERMINAL

```
$ brew install semgrep
$ brew install --cask codeql
```

04 / 05 **Karpathy skill**

Touch only what you must

`github.com/forrestchang/andrej-karpathy-skills · MIT`

Four rules in one CLAUDE.md, **built from Andrej Karpathy's notes** on how AI coders go wrong. Rule 3, **Surgical Changes**: don't refactor what isn't broken, and every changed line should trace back to your request.

The Solution

Four principles in one file that directly address these issues:

Principle	Addresses
Think Before Coding	Wrong assumptions, hidden confusion, missing tradeoffs
Simplicity First	Overcomplication, bloated abstractions
Surgical Changes	Orthogonal edits, touching code you shouldn't
Goal-Driven Execution	Leverage through tests-first, verifiable success criteria

TYPE IN CLAUDE CODE

```
> /plugin marketplace add forrestchang/andrej-karpathy-skills
> /plugin install andrej-karpathy-skills@karpathy-skills
```

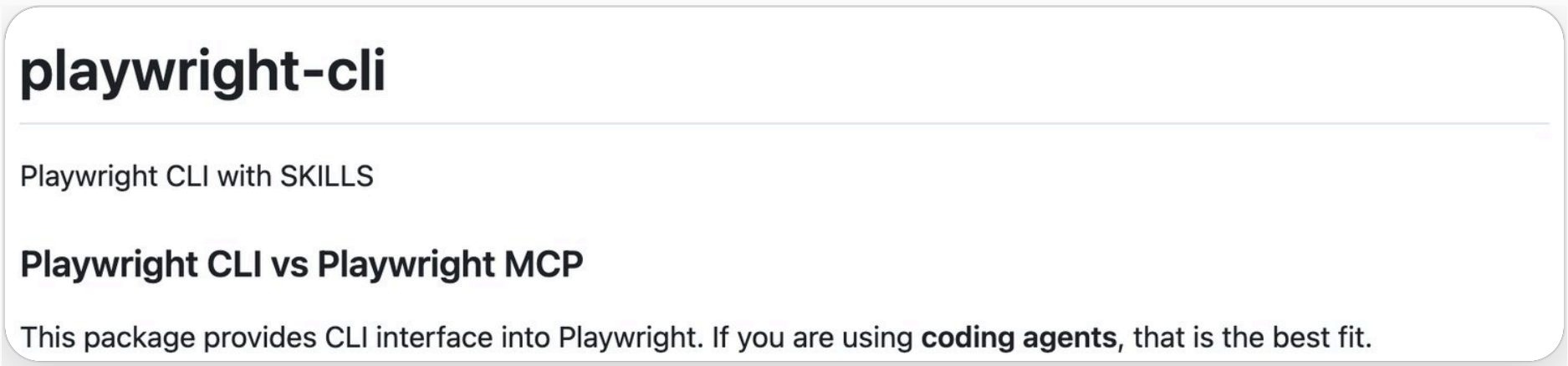
Karpathy didn't make it himself. The repo has moved to **multica-ai/andrej-karpathy-skills** and the old address still works.

05 / 05 **Playwright**

Claude checks its own work

`github.com/microsoft/playwright-cli · Apache-2.0`

Microsoft's Playwright CLI ships **skills for coding agents**. Claude opens a **real browser**, clicks through what it built, **screenshots what fails**, and can fix it.


playwright-cli

Playwright CLI with SKILLS

Playwright CLI vs Playwright MCP

This package provides CLI interface into Playwright. If you are using **coding agents**, that is the best fit.

RUN IN YOUR TERMINAL

```
$ npm install -g @playwright/cli@latest
$ playwright-cli install --skills
```

THEN ASK CLAUDE

Use playwright skills to test the signup flow on localhost:3000. Take screenshots of anything that fails, then fix it.

If something goes wrong

The new commands don't show up

Restart Claude Code after installing, so the plugins and skills load.

/plugin marketplace add fails with an SSH error

Use the full HTTPS address instead, for example `/plugin marketplace add https://github.com/trailofbits/skills.git`

Superdesign asks you to sign in

Run `superdesign login`. It needs a superdesign.dev account. Use the plugin install, not also `npx skills add`, or you get two copies (Superdesign README).

The Trail of Bits scan stops right away

The skills check for the scanners first. Install Semgrep and CodeQL (no Homebrew? `python3 -m pip install semgrep` and `gh extension install github/gh-codeql`).

Do I need all five?

No. Each one installs and works on its own.

All five in one place (1/3)

```
FIVE CLAUDE CODE SKILLS: THE INSTALL PACK

1. Superdesign (a design canvas for Claude, so your UI stops
looking like AI slop)
Type these in Claude Code, one at a time:
/plugin marketplace add superdesigndev/superdesign-skill
/plugin install superdesign@superdesign
Then run these in your terminal:
npm install -g @superdesign/cli@latest
superdesign login
It runs on superdesign.dev, so you need an account there.
Try: /superdesign:superdesign help me redesign this settings
page

2. Superpowers (asks first, writes the spec, then subagents
build, test, and review)
Type this in Claude Code:
/plugin install superpowers@claude-plugins-official
Nothing to call. It starts on its own when you begin building
something.
```

All five in one place (2/3)

```
3. Trail of Bits security skills (CodeQL and Semgrep scans, then
every finding checked)
Type these in Claude Code, one at a time:
/plugin marketplace add trailofbits/skills
/plugin install static-analysis@trailofbits
/plugin install fp-check@trailofbits
The scanners themselves install separately. Run these in your
terminal:
brew install semgrep
brew install --cask codeql
(No Homebrew? Semgrep: python3 -m pip install semgrep. CodeQL:
gh extension install github/gh-codeql, then gh codeql install-
stub)
Try: Run a Semgrep and CodeQL scan on this repo, then verify
each finding.

4. The Karpathy skill (touch only what the task needs)
Type these in Claude Code, one at a time:
/plugin marketplace add forrestchang/andrej-karpathy-skills
/plugin install andrej-karpathy-skills@karpathy-skills
It's a set of four rules built from Andrej Karpathy's notes on
AI coding. It isn't made by Karpathy himself.
```

All five in one place (3/3)

```
5. The Playwright skill (Claude opens a real browser and tests
what it built)
Run these in your terminal, inside your project folder:
npm install -g @playwright/cli@latest
playwright-cli install --skills
Try: Use playwright skills to test the signup flow on
localhost:3000. Take screenshots of anything that fails, then
fix it.

Restart Claude Code after installing, so the new plugins and
skills load.
```

This DM came from an automation

You commented on our reel and Uplika sent this guide. Uplika lets Claude post to your channels, and sends a DM when someone comments.

See how it works: uplika.com/en/integrations/claude

Connector URL: `https://api.uplika.com/mcp`