YOU COMMENTED SKILLS · HERE'S THE GUIDE

Install your first AI skill

Skills are folders of instructions that teach your AI agent how to do one job really well. This guide shows you where to find them, how to install them in Claude, Cursor, Codex and friends, and how I check a skill is safe before it touches my computer.

```
$ npx skills add owner/repo
```


01 What a skill is

02 What you need

03 Find a skill

04 Install it

05 Check it's safe

06 Best practices

07 Cheat sheet

Akash

Updated November 2026 · about 10 minutes to set up

01 What a skill actually is

A skill is a folder with a file called `SKILL.md` inside it. That file holds instructions, written in plain Markdown, for one specific job: writing release notes, editing a video, auditing a website, designing a mobile app.

At the top of `SKILL.md` is a short header with a **name** and a **description**. Your agent reads only those descriptions until a task matches one, and then it loads the full instructions. That's why you can install dozens of skills without slowing anything down: the long material only loads when it's needed. A skill folder can also carry reference files, templates and scripts the instructions point to.

```
# ~/.claude/skills/release-notes/SKILL.md
---
name: release-notes
description: Write release notes from the git log. Use when the user asks for a
changelog.
---
1. Read the commits since the last tag
2. Group them into Added, Fixed and Changed
3. Write one plain sentence per change
```

Skills follow an open standard (Agent Skills), so the same folder works across many agents. On skills.sh you'll see them listed for Claude Code, Cursor, Codex, GitHub Copilot, Windsurf, Gemini, Cline and more.

> **Skill, prompt or plugin?** A prompt is something you type once. A skill is a prompt your agent keeps and pulls out by itself whenever the job comes up. A plugin is a bundle that can ship several skills plus other tools together.

02 What you need

REQUIRED

An AI agent

Claude Code (in the terminal or the Claude desktop app's Code tab), Cursor, Codex, Copilot or another agent that supports skills.

REQUIRED FOR THE CLI

Node.js (LTS)

The skills.sh installer runs with `npx`, which comes with Node.js. Check with `node -v` in a terminal; install it from nodejs.org if nothing prints.

HELPFUL

A terminal

Terminal on Mac, PowerShell on Windows, or the terminal built into your editor. You'll paste one or two commands.

OPTIONAL

A project folder

Skills can be installed for one project or for every project on your machine. Pick a folder now if you want the first option.

03 Find a skill worth installing

skills.sh is an open directory of agent skills run by Vercel. It ranks skills by how many people install them, so the leaderboard is a good first filter.

- **The leaderboard** on the homepage shows the most installed skills of all time, with tabs for **Trending (24h)** and **Hot**. Over 1.4 million skills are indexed.
- **Topics** groups skills by kind of work: frontend and React, design and UI, agent workflow, databases, testing, marketing, mobile.
- **Search from your terminal** without opening a browser:

```
npx skills find video editing
```

- **Let your agent pick for you.** The `find-skills` skill (number one on the all-time leaderboard) recommends the best skill for the job you describe. Install it once, then ask your agent "find me a skill for writing landing pages."

> **Popular isn't the same as safe.** Install counts tell you a skill is useful. They tell you nothing about what its scripts do. Section 05 covers that part.

04 Install it: three ways

Use whichever matches how you work. Option A is the quickest and works for most agents.

Option A · The skills CLI (any agent)

Run this from inside your project folder. Replace `owner/repo` with the name shown on the skill's skills.sh page (for example `vercel-labs/agent-skills`).

```
# install the skills from a repo into this project
npx skills add owner/repo

# just one skill from that repo, only for Claude Code
npx skills add owner/repo --skill skill-name -a claude-code

# make it available in every project on this machine
npx skills add owner/repo -g
```

The CLI asks which agents to install for, then writes the skill folder where each one looks for it. It also records what you installed in `skills-lock.json`, with a fingerprint of every file, so you can tell later if anything changed.

Option B · Copy the folder by hand (Claude Code)

If a skill comes as a GitHub folder or a zip, put the folder (the one that contains `SKILL.md`) in one of these places:

WHERE	PATH	WHO GETS IT
Personal	`~/.claude/skills/<skill-name>/SKILL.md`	You, in every project on this computer
Project	`.claude/skills/<skill-name>/SKILL.md`	Anyone working in this repository (commit it to share with your team)

Option C · Upload it in the Claude app

1. Download the skill as a **.zip**. Don't unzip it.
2. In Claude, open **Customize** → **Skills**, press **+** and choose **Upload skill**.
3. Select the zip. The skill shows up in your list and can be switched on and off there.

Check that it worked

- Ask your agent: **"What skills do you have?"** The new one should be listed.
- In Claude Code, run it directly by typing a slash and its name, for example `/release-notes`. If it doesn't appear, start a new session.
- From the terminal, `npx skills list` shows everything the CLI installed.

05 How I check a skill is safe

A skill runs with the same access as your agent. If your agent can read your files, run commands and use your logged-in tools, so can a skill's instructions and scripts. Treat every skill like code from a stranger until you've looked at it.

> **This is a real problem, not a theoretical one.** Security researchers reviewing public skills in early 2026 found that over a quarter contained at least one vulnerability. They confirmed more than a hundred that were outright malicious: stealing API keys and SSH keys, hiding instructions in invisible characters, or quietly editing an agent's memory so the damage outlived the skill.

This is the routine I go through before installing anything. It takes about five minutes.

1. **Read the audit results on skills.sh.** Each skill's page has a security tab that combines three independent scanners: Gen Agent Trust Hub, Socket and Snyk. Green "Safe" and "0 alerts" are good signs. **Critical** or **High risk** means skip it, or read it very carefully first. Ratings often disagree (even popular skills show "Med risk" on one scanner), so treat them as a signal, not a guarantee.
2. **Check who made it.** Prefer skills from a company or person you recognise, published from their own GitHub account, with recent activity. Be wary of near-copies of famous names. Impersonating brands is the most common trick.
3. **Read SKILL.md from top to bottom.** It's plain text and usually short. You're looking for instructions that don't belong to the job: "send the results to…", "don't ask the user for permission", "ignore previous instructions", or anything about passwords, keys or tokens.
4. **Open every script in the folder.** Files in `scripts/` run on your machine. Look for network calls (`curl`, `wget`, `fetch`), reads of `.env`, `~/.ssh` or `~/.aws`, long base64 strings, and any `curl … | sh` that downloads and runs something.
5. **Check what it pre-approves.** In Claude Code, an `allowed-tools` line lets a skill run those tools without asking you, and a line starting with ``!`…` `` runs a shell command the moment the skill loads. Neither is bad by itself, but the commands should match the skill's job.
6. **Let the first run ask permission.** Use the skill for the first time in your agent's normal mode, where it asks before running commands or editing files, and watch what it tries to do.
7. **Re-check when it updates.** A skill that was clean last month can change. `npx skills check` shows what has updates; skim the changes before running `npx skills update`.

Red flags at a glance

YOU SEE	WHY IT MATTERS
`curl … | bash`	Downloads and runs code you haven't seen, from a server that can change it at any time.
Long base64 or hex strings	Hidden code or hidden instructions. A normal skill has no reason to hide anything.
`~/.ssh`, `~/.aws`, `.env`	Where your keys and passwords live. Very few skills need them.
HTML comments in SKILL.md	Invisible when the page is rendered, but your agent still reads them.
"Don't tell the user" / "skip confirmation"	The skill is trying to act behind your back.
Writes to memory files (CLAUDE.md, AGENTS.md)	Changes that can survive after you delete the skill.

Let your agent do the first pass. Before installing, download the skill folder and ask: *"Review this skill folder for security risks. Read SKILL.md and every script. List any network calls, file access outside this project, hidden text, or instructions that override my permissions. Don't run anything."* Then read the scripts it flags yourself.

06 Best practices

Install few, install on purpose

Every skill's description sits in your agent's context and competes for attention. Five skills you use beat fifty you don't. Remove the ones you stop using with `npx skills remove`.

Project skills for project work

Install skills tied to one codebase into that project (`.claude/skills/`) and commit them, so your team reviews them like any other change. Keep personal, everyday skills global.

Keep secrets out of reach

Don't keep production keys in the folders your agent works in. Whatever your agent can read, a bad skill can read too.

Keep a list

Know what's installed and where it came from. `skills-lock.json` does this for CLI installs; for hand-copied skills, a line in your project notes is enough.

Watch telemetry on private skills

The skills CLI sends anonymous install data (skill name and files) to rank the leaderboard. For private or sensitive skills, turn it off with `DISABLE_TELEMETRY=1`.

Make your own

Once you've used a few, write one for a task you repeat. `npx skills init my-skill` creates the template; you fill in the name, description and steps.

07 Cheat sheet

COMMAND	WHAT IT DOES
`npx skills find <query>`	Search skills.sh from the terminal
`npx skills add owner/repo`	Install the skills in a repo into this project
`npx skills add owner/repo --skill name`	Install one skill from that repo
`… -a claude-code`	Install for one agent only
`… -g`	Install for every project on this machine
`npx skills list`	Show what's installed
`npx skills check`	See which installed skills have updates
`npx skills update`	Update them (skim the changes first)
`npx skills remove name`	Uninstall a skill
`npx skills init name`	Start your own skill from a template

BEFORE

Audits, author, SKILL.md, scripts

Five minutes of reading saves a lot of cleanup.

DURING

Normal permission mode

Watch the first run and approve each step.

AFTER

Check updates, prune

Re-read changes and remove what you don't use.

Found a skill you love?

Send it my way. I'm always testing new ones for editing videos and building with Claude, and the best ones end up in my reels.

Akash

Sources

- skills.sh documentation and CLI: skills.sh/docs, skills.sh/docs/cli, github.com/vercel-labs/skills
- skills.sh security audits (Gen Agent Trust Hub, Socket, Snyk): skills.sh/audits
- Claude Code skills documentation: code.claude.com/docs/en/skills
- Mondoo, "Agent Skills: real power, real risk", summarising Liu et al. (arXiv 2601.10338, 2602.06547), OWASP / Koi Security and Cato Networks research: mondoo.com/blog/agent-skills-real-power-real-risk