research · 2026-08-30

What agent skills are actually made of

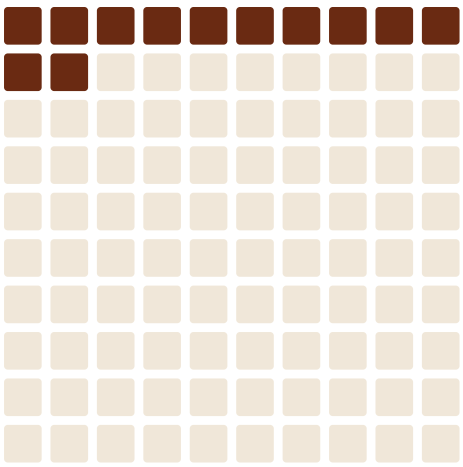
Plicara Research · 2026-08-30 ·

An agent skill is a folder holding instructions for an AI coding assistant, written in ordinary prose. Anthropic introduced the format in October 2025 and it is now an open specification that around forty products read. These eight figures ask one question of the 1.9 million of them on GitHub: when a skill does come with code attached, what language is that code in?

Most skills are just writing

A skill is a folder: a SKILL.md file of instructions, plus whatever its author puts beside it. **Coming with code means that folder holds a file the agent can run**, so `pdf-tools/SKILL.md` sitting next to `pdf-tools/scripts/extract.py` counts, and a skill that only describes what to do does not. It is the skill's own folder that matters, not the repository around it, which is usually a software project full of code either way. Each square here is one skill in a hundred: 12 come with code, the other 88 are instructions and nothing else. The bar underneath breaks down the 5.9 million files that do sit beside a SKILL.md, and 50% of those are more writing rather than code. An independent study of 31,132 skills from two marketplaces found almost the same rate, 11.5% against our 11.78%.

FIGURE 1. UNIT CHART, AND THE COMPOSITION OF EVERYTHING SKILLS BUNDLE



12

in every 100 skills
ship any code at all

The other 88 are prose:
instructions for a model,
with nothing to execute.

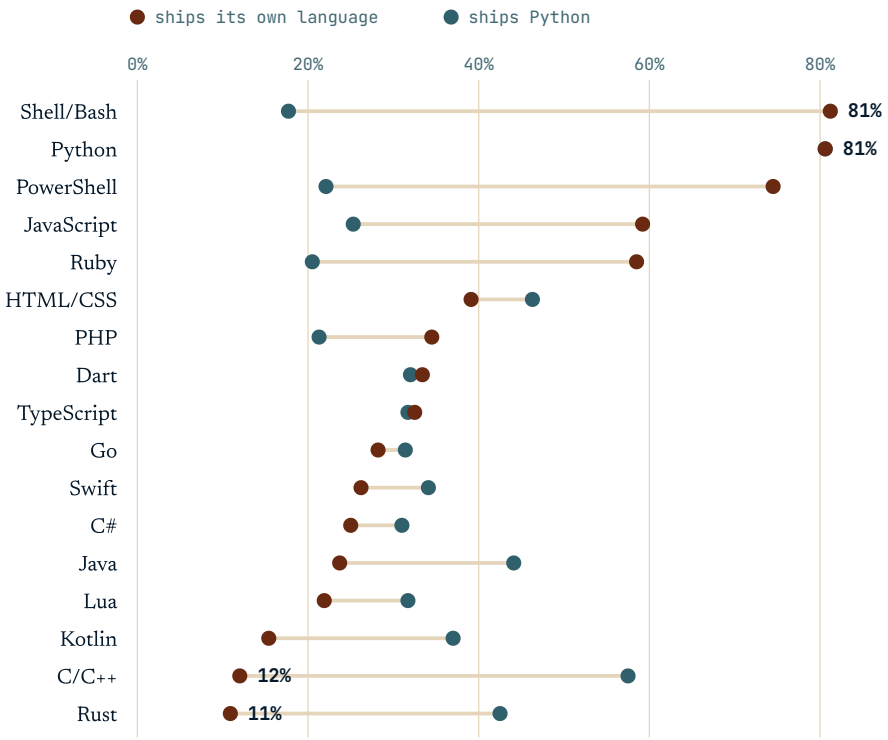
and of the 5,855,080 files they do bundle

	SKILLS	SHARE
bundle nothing at all	1,195,708	63.67%
ship code	218,120	**11.78%**
ship code, counting root-level skills	236,368	12.59%

A Rust project's skill is usually written in Python

Each row is a group of repositories, sorted by the language GitHub says the project is mainly written in. The orange dot shows how often those projects' skills contain code in that same language; the blue dot shows how often they contain Python instead. Reading down the list the two dots trade places: a Shell project writes its skills in Shell 81% of the time, but a Rust project writes them in Rust only 11% of the time and reaches for Python 42%. Nothing here reads a word of the skill text, so it is an independent check on the same idea. It matches what a study of how language models pick languages found from the other direction: asked to start high-performance projects, models chose Python 58% of the time and Rust not once.

FIGURE 2. DUMBBELL CHART, THE REPOSITORY'S OWN LANGUAGE AGAINST PYTHON

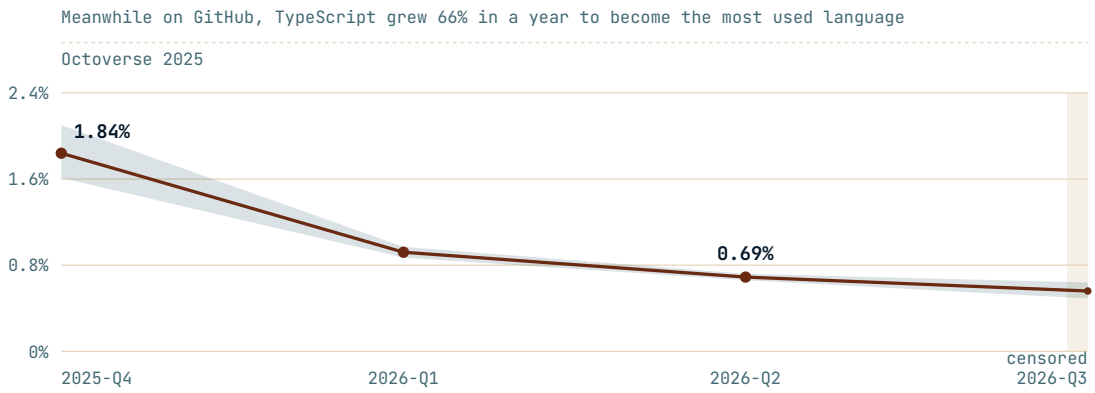


REPOSITORY IS MOSTLY	SKILLS	OWN LANGUAGE	PYTHON
Shell/Bash	13,577	**81.2%**	17.7%
Python	129,656	**80.6%**	80.6%
JavaScript	14,920	**59.2%**	25.3%
TypeScript	29,213	**32.5%**	31.7%
Java	1,075	**23.7%**	44.1%
C/C++	1,307	**12.0%**	57.5%
Rust	2,928	**10.9%**	42.5%

TypeScript is growing everywhere except here

The line is the share of newly written skills carrying at least one TypeScript file, quarter by quarter, and the shaded band around it is the margin of error. It falls the whole way, from 1.84% to 0.56%. The note along the top is GitHub's own count of the opposite: Octoverse 2025 reports TypeScript passing Python in August 2025 to become the most used language on the site, on 66% growth in a year. The two count different things, contributors there and files here, so this is one measure falling while another rises rather than a contradiction. The final quarter is greyed out because collection stopped partway through it, so it is shown but never compared.

FIGURE 3. SHARE OF NEW SKILLS SHIPPING TYPESCRIPT, BY QUARTER

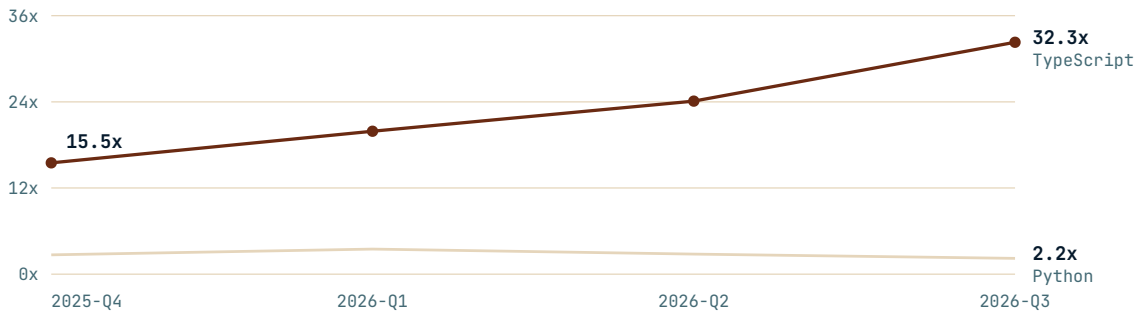


QUARTER	NEW SKILLS	MENTION IT	SHIP IT
2025-Q4	11,439	28.43%	**1.84%**
2026-Q1	147,837	18.34%	**0.92%**
2026-Q2	254,656	16.63%	**0.69%**
2026-Q3	41,490	18.07%	**0.56%**

The distance between talking and writing keeps growing

Both halves of this count skills. Take the skills written in one quarter, count how many name a language anywhere in their text, then count how many actually hold a file in it, and divide. In the last quarter 18% of new skills mentioned TypeScript while 0.56% contained a `.ts` file, and 18 divided by 0.56 is the 32.3 at the right-hand end. A value of 1 would mean people write what they talk about. TypeScript climbs from 15.5 to 32.3, so the gap widens every quarter, while Python is the faint line along the bottom holding near 2.2.

FIGURE 4. MENTION-TO-SHIP RATIO BY QUARTER, TYPESCRIPT AGAINST PYTHON

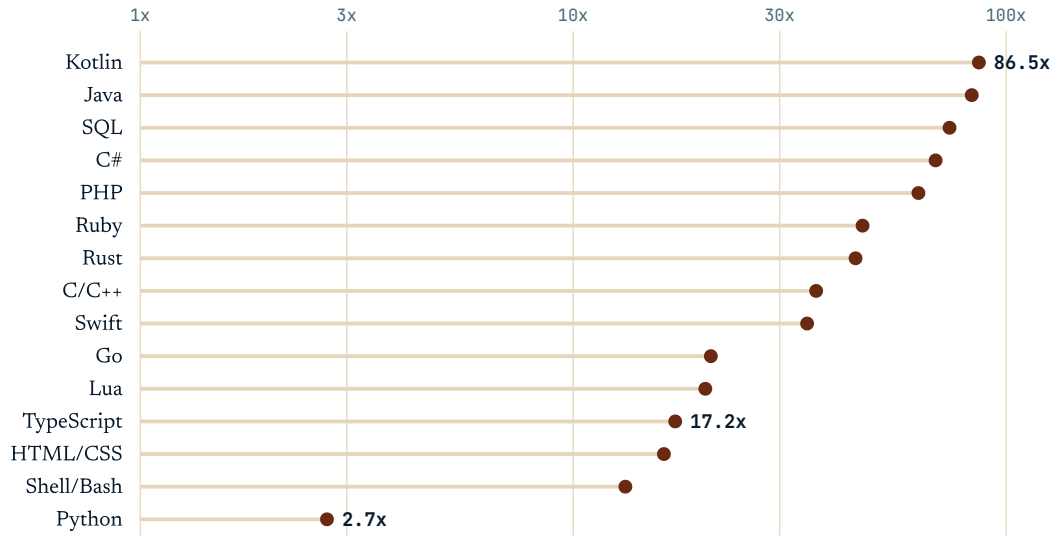


QUARTER	TYPESCRIPT	PYTHON
2025-Q4	**15.5x**	2.7x
2026-Q1	**19.9x**	3.5x
2026-Q2	**24.1x**	2.8x
2026-Q3	**32.3x**	2.2x

Some languages are only ever discussed

The same ratio as the previous figure, one row per language, over the whole corpus rather than by quarter. **Both numbers count skills, not repositories**: how many skills name the language, divided by how many skills hold a file in it. Kotlin sits at the top, named in 15,140 skills and present as a file in 175, which is 86 times more talk than code. Python sits at the bottom at 2.7x, near enough to 1 that the people who mention it mostly go on to write it. The scale stretches as it moves right, so each gridline is about three times the one before.

FIGURE 5. MENTION-TO-SHIP RATIO PER LANGUAGE, LOG SCALE

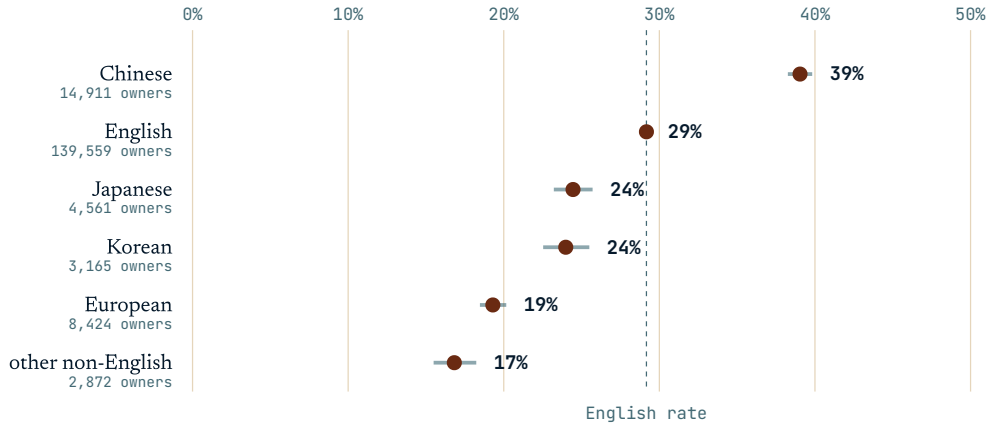


LANGUAGE	MENTION IT	SHIP IT	TIMES MORE TALK
Kotlin	15,140	175	**86.5x**
Java	45,086	541	**83.3x**
SQL	105,483	1,426	**74.0x**
C#	51,330	747	**68.7x**
PHP	27,213	434	**62.7x**
Ruby	18,252	392	**46.6x**

Chinese-language skills carry code twice as often

Each row groups skills by the human language they are written in, then asks what share of the people publishing them ever include code. The dot is that share, the bar through it is the margin of error, and the dotted line marks the English rate so the comparison is visible rather than arithmetic. Chinese sits well to the right at 39% against English at 29%. Every other group sits to the left of the line, so this is not a general non-English effect: it is specific to Chinese authors. The language column is the one the previous article in this series built.

FIGURE 6. SHARE OF OWNERS SHIPPING CODE, BY THE LANGUAGE THE SKILL IS WRITTEN IN

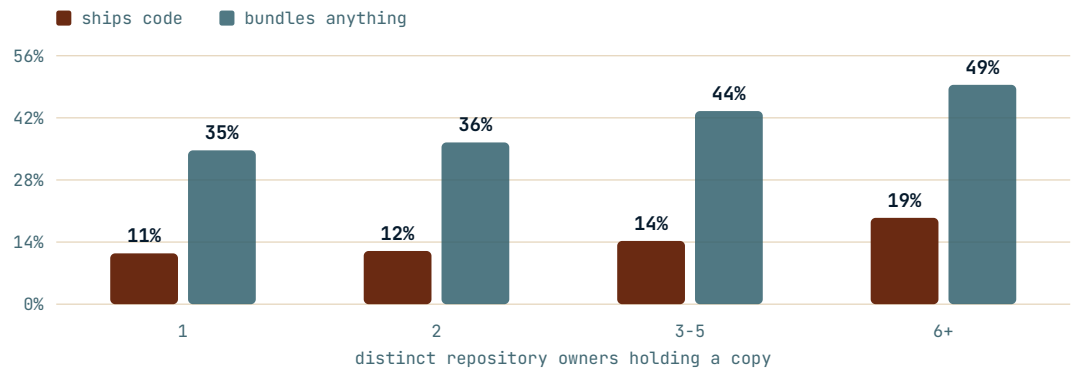


WRITTEN IN	OWNERS	EVER SHIP CODE	MARGIN OF ERROR
Chinese	14,911	**39.04%**	38.26 to 39.82%
English	139,559	**29.17%**	28.93 to 29.41%
Japanese	4,561	**24.45%**	23.22 to 25.71%
Korean	3,165	**23.98%**	22.53 to 25.5%
European	8,424	**19.3%**	18.47 to 20.16%
other non-English	2,872	**16.82%**	15.49 to 18.23%

The most-copied skills are the ones with code in them

Skills are grouped by how many separate people hold a copy of the identical file, from never copied on the left to six or more owners on the right. Orange is the share whose folder holds a runnable file, in the sense set out in the first figure; blue is the share holding any extra file at all, code or not. Both rise as you move right, from 11% to 19% on code, but the jump is at the far end rather than a steady climb. Whether the code is what makes them worth copying, or popular skills simply attract more work, is not a question a file listing can settle.

FIGURE 7. SHARE SHIPPING CODE, BY HOW MANY OWNERS HOLD A COPY



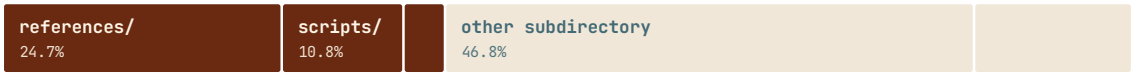
PEOPLE HOLDING A COPY	SKILLS	SHIP CODE	BUNDLE ANYTHING
1	1,622,275	**11.45%**	34.68%
2	118,683	**12.0%**	36.46%
3-5	67,409	**14.25%**	43.52%
6+	43,721	**19.47%**	49.44%

The official folder layout is a minority habit

The specification sets out three folders for a skill's extra files: scripts/, references/ and assets/. This bar shows where those 5.9 million files actually sit, and the highlighted stretch on the left is those three folders put together. They account for 39% of everything. The rest sits in folders people invented themselves, or loose beside the skill file with no folder at all.

FIGURE 8. WHERE BUNDLED FILES SIT, AGAINST THE LAYOUT THE SPECIFICATION DEFINES

the three directories the specification names: 39% of files



WHERE THE FILE SITS	FILES	SHARE
other subdirectory	2,742,463	**46.8%**
references/	1,446,376	**24.7%**
loose beside SKILL.md	819,824	**14.0%**
scripts/	630,900	**10.8%**
assets/	215,517	**3.7%**

what this does not show

Every "ships code" figure is a floor and a ceiling at once. A floor because the crawler truncated the folder listing for 13.43% of skills, so some code went uncounted. A ceiling because a SKILL.md at the root of a repository has the whole project sitting beside it, and those skills, 25,893 of them, report code at 70% when the corpus as a whole reports 11.78%. They are excluded from every figure here, which is why the headline is 11.78% rather than 12.59%.

The two figures that move through time rest on the minority of skills carrying commit history, and on the first commit that touched that copy rather than the first appearance of the content anywhere. A crawl also sees only survivors, so a skill created and deleted before July 2026 is invisible here.

Naming a language and shipping a file in it are both crude. The mention patterns are deliberately loose, so the mention counts are ceilings, and a file extension says what a file is rather than whether it runs or matters.

credit

None of this exists without the dataset, which was built and released by someone else, and all credit for collecting, deduplicating and documenting 3.8 million skill files belongs to its authors:

> Giuseppe Destefanis, Daniel Graziotin, Matteo Vaccargiu, and Marco Ortu. 2027. GitSkills: A Dataset of Agent Skills on GitHub. In *Proceedings of the 24th International Conference on Mining Software Repositories (MSR '27)*.

Preprint arXiv:2608.10906, archive 10.5281/zenodo.21875637, Parquet mirror `mvaccargiu/gitskills`, licence CC BY 4.0. We are not affiliated with its authors, with MSR, or with the Mining Challenge, so nothing here should be read as endorsed by them: the dataset is theirs, and the analysis and any error in it is ours. This article answers one question the authors pose and leave open, how often skills bundle executable files and how widely those skills are copied.

Analysis code lives at github.com/plicara/articles under `gitskills-analysis/`, where every figure and every number in the sentences above is generated from a single machine-readable export and never typed by hand, so the whole thing can be regenerated and checked.

Found something wrong? We would genuinely like to know.

sources

- GitHub Octoverse 2025. TypeScript reaching number one on GitHub by monthly contributors in August 2025, up 66% year over year, which GitHub attributes partly to agent-assisted coding. Cited in the third figure.
- Twist et al., A Study of LLMs' Preferences for Libraries and Programming Languages. Language models choosing Python for 90 to 97% of benchmark tasks, and not choosing Rust once on high-performance ones. Cited in the second figure.
- Liu et al., Agent Skills in the Wild. The 31,132-skill marketplace study whose script-bundling rate the first figure replicates.
- The Agent Skills specification. The required front matter and the three optional folders the last figure measures against.
- What language are agent skills written in?. The previous article in this series, whose language column the sixth figure reuses.