Claude Writes 80% of Anthropic's Merged Code; Practical Agent Loops Take Shape

Coding Agents Alpha Tracker

2026-06-05

Claude Writes 80% of Anthropic's Merged Code; Practical Agent Loops Take Shape

By Coding Agents Alpha Tracker • June 5, 2026

Anthropic shared internal metrics showing Claude now writes most merged code, while practitioners detailed the loops that make agents useful in real workflows. This brief focuses on copyable self-review, issue triage, prototyping, flaky-test debugging, and the newest releases from Cursor, LangChain, Codex, and Cognition.

TOP SIGNAL

Anthropic's Alex Albert shared the clearest production datapoint of the day: Claude now writes **>80%** of code merged into Anthropic's codebase, many researchers haven't hand-written code in months, engineers ship **8×** more code than in 2024, and open-ended engineering-task success rose from ~26% to **76%** in six months [1]. In Matthew Berman's recap of Anthropic's talk, the median respondent estimated roughly **4×** more output with Mythos Preview, and he points out that human review becomes the bottleneck if code generation outruns review speed [2].

TRY THIS

- **Peter Steinberger's `vision.md` issue sweeper.** 1) Write a `vision.md` with project goals, invariants, and explicit "want / don't want" rules. 2) On every new issue or PR, trigger Codex from a GitHub Action. 3) Have the agent read `vision.md` and either comment on or close the item. 4) Re-run the sweep weekly across open issues and PRs. Steinberger says this closed ~15k issues on his open-source projects, which fits his broader rule: help the agent close the loop autonomously [3].

- **Make Codex review Codex before you land a PR.** Steinberger adds a one-line rule in `agents.md`: `before you commit or land a PR, if you haven't done auto review, run AutoReview and review again`, letting Codex call itself for multiple review/fix rounds. Put project invariants in `agents.md`, and periodically ask the agent to rewrite its own instructions or flag confusing sections; Theo's comparison is a useful reminder not to force one steering file across models—he keeps a much longer `Claude.md` because he steers Claude differently from OpenAI models [3, 4].
- **Simon Willison's prototype-first API loop.** Start with `review the last commit`, then ask the agent to `brainstorm a prototype. Three features against that new API`; run it in a branch or worktree, test the throwaway prototype yourself, then feed the verified artifact back into production with a prompt like `add a paste file feature based on the prototype in File Paste HTML` [5].
- **Stop tolerating flaky tests.** Willison's move: tell the agent `you've got Docker; try and reproduce this thing` when CI fails in a Linux or Python variant that doesn't fail locally, and let it reproduce the environment before diagnosing the bug [5]. When the code path is important enough to inspect, switch into his "active refactoring" mode with prompts like `refactor the test to reduce duplicate code, rename variables, [ensure] consistency with this other file` or `explain it and add comments` [5].

WHAT SHIPPED

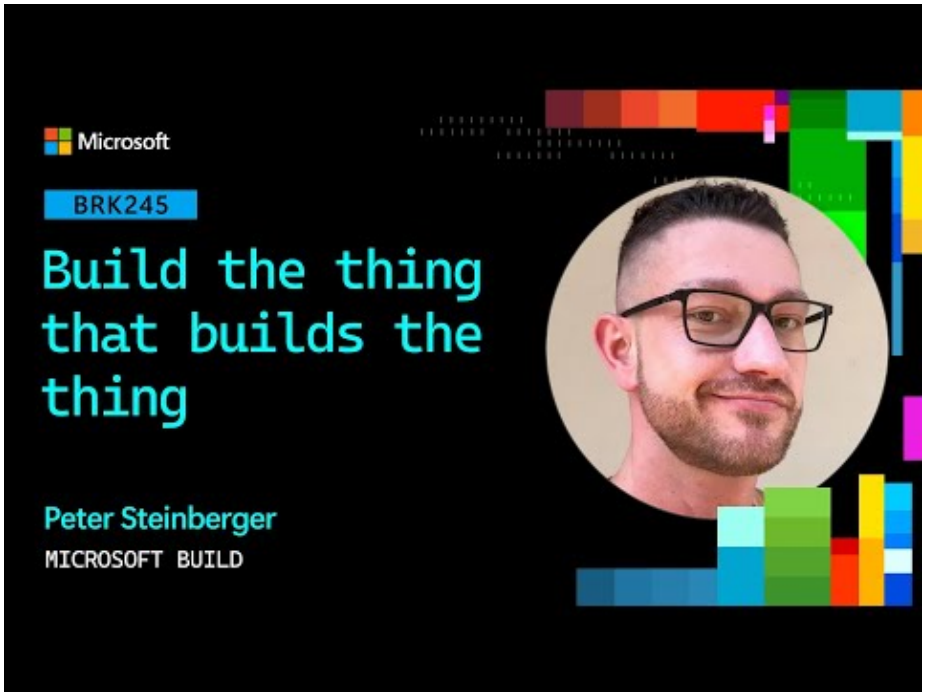
- **Cursor canvases** — New context explorer breaks down token use across system prompt, tool definitions, rules, skills, and more; Design Mode lets you select and annotate UI elements directly; canvases can now be published and shared via URL. Changelog: cursor.com/changelog/canvas-improvements [6, 7, 8, 9]. Some users are already calling the publish flow "Cursor Sites" [10].
- **LangSmith Engine** — LangChain is packaging the standard agent-improvement loop — `Trace → find failure patterns → fix prompts or code → create evals → test → ship → repeat` — and says Engine turns production traces into named issues, root-cause analysis, proposed fixes, and stronger eval coverage. June 11 walkthrough: events.langchain.com/webinar/how-to-shorten-the-path-with-langsmith-engine/ [11, 12].
- **LangSmith Sandboxes** — GA's new Sandbox CLI can build snapshots from Dockerfiles, manage sandboxes, open interactive consoles, tunnel raw TCP, and expose sandboxes to `ssh`, `scp`, `rsync`, and `sftp` like a normal Linux box. Blog: langchain.com/blog/langsmith-sandboxes-generally-available [13].
- **Codex Python SDK** — OpenAI's programmatic Codex entry point is live via `pip install openai-codex`; docs: developers.openai.com/codex/sdk#python-

library [14].

- **Fleet's boring-agent win** — LangChain says one of its first internally adopted Fleet agents, `@docs_plz`, takes a docs request in Slack, opens a ticket, and puts up a PR; Brace Sproul says docs shipping velocity "skyrocketed" after rollout. Product link: fleet.langchain.com [15, 16, 17].
- **Cognition's long-horizon evals** — Devin's first public long-horizon eval covers real enterprise Java, TypeScript, Python, and C# feature work, bugfixes, and migrations using 258 sessions from 126 users; swyx contrasts its **up to 100-hour** task horizon with METR's ~16-hour cap, and scaling01 argues the benchmark may saturate quickly unless task distribution changes [18, 19].

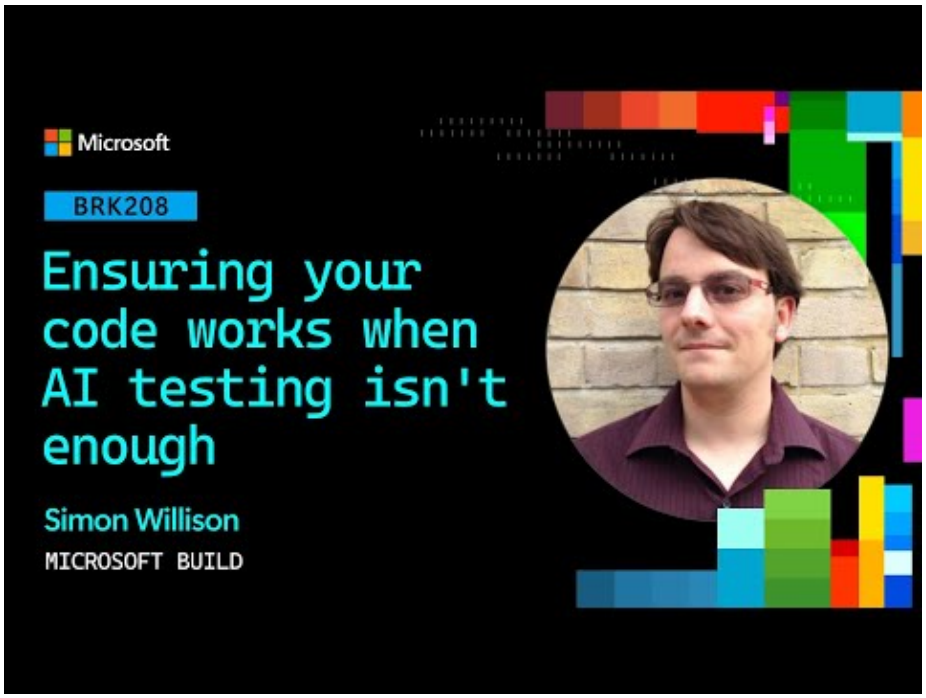
GO DEEPER

- **22:01-25:33 — Peter Steinberger on Crabbox.** A practical walkthrough of remote test execution on cloud VMs, cross-platform runs, VNC, and screenshot/click/type tools so an agent can do its own end-to-end verification instead of stopping at unit tests [3].



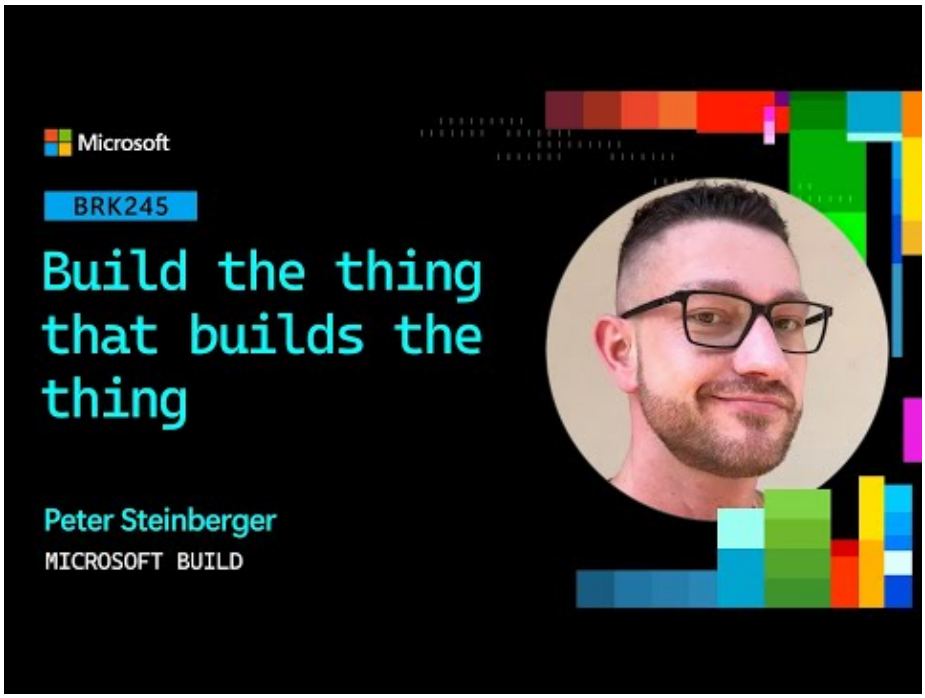
Build the thing that builds the thing | BRK245 (22:01)

- **21:58-24:20 — Simon Willison on sandboxing.** Useful if you're letting agents run generated code: he walks through CSP, sandboxed iframes, and WebAssembly/WASI, then explains why he prompts agents to try to escape the sandbox as a test [5].



Ensuring your code works when AI testing isn't enough | BRK208 (21:58)

- **27:46-30:16 — Peter Steinberger on AutoReview.** Good compact explanation of the "Codex calls Codex" pattern, plus why invariants belong in `agents.md` before you trust auto-review loops [3].



Build the thing that builds the thing | BRK245 (27:46)

- **Guide — custom harnesses.** If you're building your own agent runtime, LangChain's harness guide is worth a read because it states the job plainly: get the model the right context at the right time for the task [20]. langchain.com/blog/how-to-build-a-custom-agent-harness

Editorial take: the highest-leverage work now sits around the agent — better context, better invariants, better self-review, and better sandboxes — not just better prompting [20, 3, 5].

Sources

1. X post by @alexalbert___
2. Anthopic did a thing…
3. Build the thing that builds the thing | BRK245
4. X post by @theo
5. Ensuring your code works when AI testing isn't enough | BRK208
6. X post by @cursor_ai
7. X post by @cursor_ai
8. X post by @cursor_ai
9. X post by @cursor_ai
10. X post by @jediahkatz
11. X post by @LangChain

12. X post by @LangChain
13. X post by @LangChain
14. X post by @thsottiaux
15. X post by @BraceSproul
16. X post by @LangChain
17. X post by @LangChain
18. X post by @swyx
19. X post by @scaling01
20. X post by @LangChain