Long-Running Agent Loops, GPT-5.5 in Production, and the New Repo-Hardening Playbook

Coding Agents Alpha Tracker

2026-05-23

Long-Running Agent Loops, GPT-5.5 in Production, and the New Repo-Hardening Playbook

By Coding Agents Alpha Tracker • May 23, 2026

The strongest signal today is operational: practitioners are giving coding agents bounded milestones and enough runway to handle repo hardening, browser-driven training jobs, and real production code. Inside are the most copyable workflows, the tool and skill releases that matter, and the clips and repos worth studying next.

TOP SIGNAL

- **Long-horizon agents are finally doing the boring, high-value work.** swyx's Kakuna flow is simple: run `/plan`, then `/goal`, and let the agent spend ~16 hours / 103 commits hardening a fragile MVP without changing product behavior; reach_vb used the same pattern with Codex, giving it a screenshot plus `/goal`, and the agent drove a signed-in Colab session via Chrome, handled runtime weirdness, launched a T4 training job, and finished with 99/100 exact random checks [1, 2, 3]. The real shift is milestone ownership, not autocomplete: OpenAI is formalizing that with `/goal`, but DHH still says AI-written production code needs review, and Armin Ronacher's Clanker example shows why—a 10-line intent can still explode into a 300-line diff when the agent edits the wrong layer [4, 5, 6, 7].

TRY THIS

- **Run a hardening pass before you add more features.** swyx's Kakuna pattern is: `1)` start with `/plan`, `2)` switch to `/goal`, `3)` let it run

for a day, `4)` review the self-audit and verify behavior stayed the same. The reported outcome was the same app back, but with the boring work done—tests, maintainability work, and subagent-parallelized cleanup that made the repo easier to build on long term [1, 2].

- **Use UI context + a milestone when the work leaves the editor.** reach_vb's minimal setup for Codex was a screenshot plus `/goal`; Codex then operated Colab through Chrome and babysat the full run. OpenAI's docs and Google's Anti Gravity team describe the same deeper pattern: give the agent a specific milestone or JIRA ticket, let it run, use side chats/check-ins to inspect progress, and only step in when it needs information that is not already written down [3, 4, 8]. For risky actions, keep explicit confirmations on until trust is earned [9].
- **Make the app agent-testable on day one.** Google's Anti Gravity team recommends designing new apps so the agent can boot the app, click through flows, and turn those traces into Playwright-style integration tests. If you wait until later, they explicitly say existing products may need re-architecture before agents can test them cleanly [8].
- **If you build agent tooling, validate edits in the harness, not in the prompt.** Salvatore Sanfilippo's progression went from classic old/new replacement to line tags, then to whole-file CRC tags, and finally to a cleaner harness design where read/search remembers the last-seen lines and edit calls either fail or force a reread if those lines moved. That directly addresses line-offset drift and duplicate-occurrence failures [10].

WHAT SHIPPED

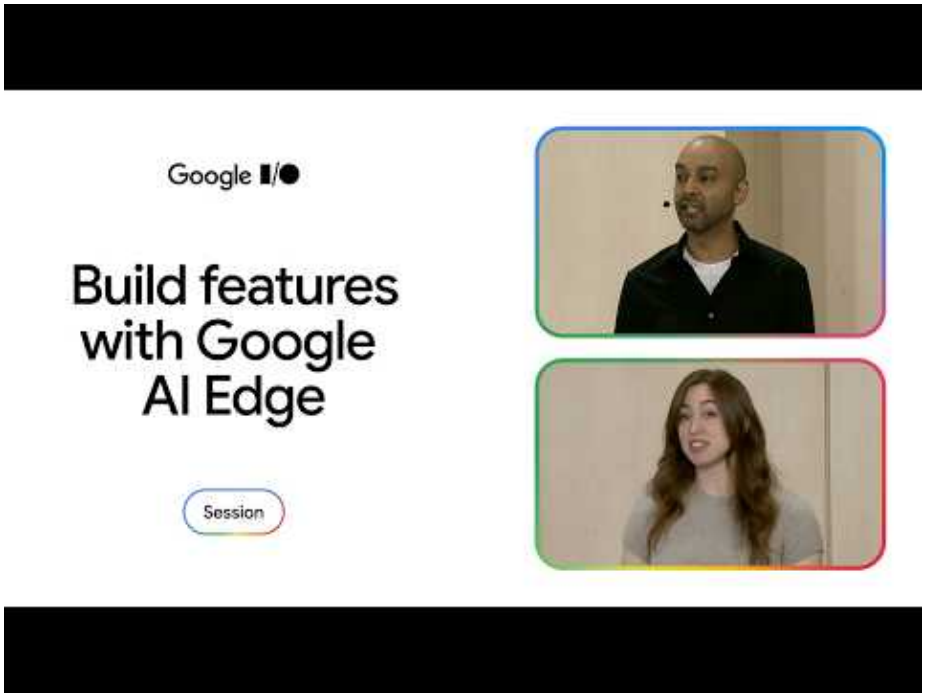
- **GPT-5.5 looks materially stronger on hard agent work.** DHH says it now beats Opus 4.7 for complicated agent tasks after GPT-5.2 lagged badly; in Omarchy 4, GPT-5.5 wrote the majority of 30,000 new lines, especially QML, and he still stresses review. He also says it is unusually good at explaining his own subtle Basecamp JavaScript. Study: Omarchy PR #5856 [11, 5, 12].
- **Cursor SDK is live.** You can now build custom agents with Composer 2.5 in Python and TypeScript, with docs at cursor.com/docs/sdk/python. Cursor is also discounting Composer usage in the SDK by 90% for the long weekend [13, 14, 15].
- **Kakuna is a new open-source hardening skill worth watching.** swyx describes it as checklists that only harden codebases, with subagent parallelism and strong opinions about agent-friendly repo design. Repo: swyxio/skills#kakuna-codebase-hardening-suite [1, 16].
- **The iOS app builder SKILL went public for any agent.** Riley Brown says the package lets agents build Swift iOS apps and get them onto a phone's Home Screen; published resources are SKILL.md and the CLI

package. The listed agent support includes ChatGPT + Codex remote, Hermes, Openclaw, Cursor/Lovable/Replit, and Claude Code [17, 18].

- **T3 Code's remote workflow looks polished.** Theo says it is two clicks to get a URL for remote worktrees on a Mac Mini with Tailscale built in; he also says the product is built on OpenAI's harness, with OpenAI actively supporting development [19, 20].
- **Review-loop skills are getting packaged.** steipete's `codex /review`-until-clean skill is now moving into openclaw/agent-skills; his caveat is the right one—this cleans up issues, not system architecture [21, 22].
- **Pi + Cursor models got a tighter bridge.** Ben Tossell one-shotted a droid SDK in ~5 minutes with Composer 2.5 Fast inside Pi; the `pi-cursor-sdk` update adds Cursor models with native capabilities plus Pi extensions/tools through an MCP bridge. Repo: pi-droid-sdk [23, 24, 25].

GO DEEPER

- **45:24-46:20 — Anti Gravity on agent-generated integration tests.** Best timeless pattern in today's set: make the agent able to launch the app, click through it, and turn those traces into Playwright tests. They explicitly say retrofitting existing products means re-architecting pieces [8].



The future of software development (45:24)

- **06:06-06:32 — Assign a JIRA ticket is the cleanest long-run mental model.** Anti Gravity's enterprise lead describes a set-it-and-forget-it loop where chat becomes a retrieval and unblocking channel instead of the place where work happens [8].
- **41:54-43:50 (+ 46:02-47:03) — Cogent on hot vs cold context.** Best naming scheme in today's material: *hot* context is actively used and can stay alive across handoffs; *cold* context is archived/indexed and accumulated by background processes. That is a reusable memory pattern for any long-running agent system [26].
- **12:35-13:35 — Thibault Sottiaux on where pure vibe coding still breaks.** Fine for experiments and joy projects; if you are targeting serious scale, keep a technical owner in the loop until agents get much better at long-term maintainability [27].
- **Repos worth studying.**
  - Omarchy PR #5856 — public 30,000-line AI-heavy conversion work on a real codebase, with the author explicitly saying review is still required [5]
  - Kakuna codebase hardening suite — one of the clearest public examples of packaging the same app with a more maintainable repo into a reusable skill [1, 16]
  - codex-review SKILL.md and openclaw/agent-skills — small, practical references for turning review loops into reusable skills while keeping humans responsible for architecture [21, 22]

Editorial take: today's edge is not more codegen—it is giving agents a bounded goal, a harness that catches drift, and a review loop that keeps architecture debt from compounding [4, 10, 21, 7].

Sources

1. X post by @swyx
2. X post by @swyx
3. X post by @reach_vb
4. X post by @OpenAIDevs
5. X post by @dhh
6. X post by @mitsuhiko
7. X post by @mitsuhiko
8. The future of software development
9. Google I/O 2026 Recap with Logan Kilpatrick, Josh Woodward and Tulsee Doshi
10. La programmazione è ancora interessante

11. X post by @dhh
12. X post by @dhh
13. X post by @cursor_ai
14. X post by @cursor_ai
15. X post by @sualehasif996
16. X post by @swyx
17. X post by @rileybrown
18. X post by @anshnanda
19. X post by @theo
20. X post by @theo
21. X post by @steipete
22. X post by @steipete
23. X post by @bentossell
24. X post by @fitchmultz
25. X post by @bentossell
26. Inside Cogent's three-agent architecture for autonomous defense | Geng Sng (Co-founder, Cogent)
27. Head of ChatGPT & Codex: agents for normal people are HERE