

Long unattended agent runs: Willison’s overnight Ultracode rebalance and Steinberger’s 600-PR migration with a planner, a coder and a challenger

Coding Agents Alpha Tracker

2026-10-21

Long unattended agent runs: Willison’s overnight Ultracode rebalance and Steinberger’s 600-PR migration with a planner, a coder and a chal- lenger

By Coding Agents Alpha Tracker • October 21, 2026

Two practitioners show multi-day, mostly autonomous agent work and how they set it up. Also: verification paths for agents, AGENTS.md as a scheduler for follow-up work, OpenAI’s longer cache retention, and a large T3 Code Stable release.

Long runs: give a blunt goal, then a structured brief

Simon Willison built *Battle Drones*, a browser RTS, mostly with Claude Code. Two of his prompts are worth reusing.

- **A vague goal plus orchestration.** He sent Opus 5.5 the prompt “This game isn’t fun enough yet. Make it a LOT more fun.” in Ultracode mode, where Claude Code runs “dynamic workflows”: JavaScript scripts that orchestrate many subagents at once [1]. The run took from evening until late the next morning [1]. Claude ran more than a thousand simulated playtests and used the results to rebalance units and add features. In 1,200 AI-vs-AI matches, games still undecided after 25 minutes fell from 14% to 0.9% [1].
- **Research and critique before building.** When the game’s AI ignored infantry and troop carriers, he wrote a detailed design brief: the incentives, the map constraints and the win condition. It ended with “First research

and then critique this design. Only build after we have agreed on any modifications” [1]. The result was the Archipelago map, which he calls the most fun map in the game [1].

Other lessons from the project:

- Static HTML/JS with Three.js is a strong medium for agent-built games because the models know those platforms well [1].
- Giving Claude an OpenAI API key lets it generate images, look at them and refine its prompts [1].
- Replays and multiplayer need a deterministic simulation. Claude had to replace `Math.sin()`/`Math.cos()` with custom versions because Chrome and Safari returned slightly different numbers [1].

Peter Steinberger started a loop to move OpenClaw’s Lit frontend to Solid 2.0. Removing Lit took about 6 days, and making the code idiomatic Solid took another week. He checked in every other day, and the work took about 600 PRs [2]. The agent missed a few small regressions, mostly button shapes and minor UX details [3]. His setup:

- **Astra plans and Sol 6.1 codes.** Prompting Astra to run Codex CLI in worktrees worked “FAR better than relying on subagents” [4].
- **A second agent challenges the first.** Instead of one agent checking its own progress against a text file, as with `/goal`, a separate agent pushes back on Astra. He built this as an OpenClaw plugin he calls “smart goal” [4].

He also moved OpenClaw to `tsrs` (`npx tsrs`). Its announcement describes a Rust port of the TypeScript compiler, built in collaboration with the TypeScript team. It claims up to 8x the speed of TS 7.0 and about 70% less memory, runs in WASM, and is due to join Oxc [5]. Steinberger calls it a drop-in replacement and saw about a 5x speedup [6], or 9x against stable releases, since they were already on 7.1-dev [7].

Give agents a paved path to verification

Kent C. Dodds says the fix for low confidence in legacy codebases is verification. Agents should do what humans did: run with mock data, use the UI, seed varied database data and deploy to a production-like environment. They will do this reliably if you give them a paved path [8]. His agents maintain a `control-kody` CLI for exactly this (SKILL.md) [8]. Separately, a team that was “all in” on agents turned out not to be running tests in CI. Dodds calls that a prompting problem: an agent asked to audit the codebase for obvious problems would have caught it [9].

Theo reported something similar after a rough night. Most breakages came from legacy vendors, GitHub Actions and flaky Chromium-driver tests. The “slop” parts, such as `ts-rust`, were stable [10]. In a new video, he describes building his machine images for the agents rather than for himself. The image

is a customized, headless Ubuntu with Bash instead of Zsh. His audit found over 30% of Claude’s errors came from Zsh’s newline handling breaking ripgrep commands, and agents also did worse on Arch and CachyOS dev boxes [11].

AGENTS.md as a scheduler for follow-up work

After the eMudhra CA key leak, **Geoffrey Huntley** asked an agent whether his NixOS fleet was affected. Because Nix has a single source of truth, one patch covers transitive dependencies: patch, commit, autodeploy [12]. His new trick is to have the agent write a dated reminder into AGENTS.md so the temporary workaround gets backed out later [12]. The reminder names the nixpkgs commit to check for, the command to run, a recheck every two days, which files to delete, and which unrelated changes to keep [12].

Overalls also got “hive mode.” Agents in the same profile zone (oss, standard or customer) share a message board for coordinating work. The customer zone has no outbound connectivity, and a machine’s entries are wiped when it suspends [13]. Huntley points to Claude Code’s `crossSessionInbound=accept` setting as the apparent primitive behind hive swarms [14], and says it has him “back to loving claude” [15]. He is moving 95% of his usage to Anthropic because OpenAI “falls over” after every usage reset [16].

OpenAI: longer cache retention and a usage-limit fix

The API now offers 12-hour prompt caching, so persistent agents pay the cached-input rate even when requests are hours apart [17]. Tibo says logged-in Codex and Work users get cache kept “forever,” with guaranteed hits when they resume old conversations [18]. OpenAI also says about 40,000 accounts had usage limits that didn’t match their plan. It has corrected the issue and will add credits for affected users [19].

T3 Code Stable

The new T3 Code Stable release has more than 1,400 changes [20]. Highlights:

- Add a project to another machine from the “Run on” menu.
- PR diffs that separate code, tests and docs.
- Idle Claude threads compact early so the cache stays warm (beta).
- `t3 connect` reachability checks.
- Webhook-task filters and dedupe.
- More reliable skill discovery [20].

Theo suggests moving back to Stable because Nightly is getting more experimental [20].

Other notes

- **Tenex’s “meta harness”** is structured repo docs plus a CLI that agents use to drive plans and readouts. Agents log their decisions, and everything is checked into Git, so the reasons behind changes live in the repo [21]. Tenex has loosened the plan template as models improved [21]. Its engineers mostly drive with Opus 5.5 and use Codex to review, which they say gives the best results [21]. They report output roughly doubled this year [21].
 - **Claude Code Projects tip:** tell the coordinator to save a project-memory rule to post a one-line update linking the thread whenever a thread is blocked or finishes a milestone. Before this, it wouldn’t even report blocked threads [22].
 - **Kody** lets agents file feedback about their experience, which often becomes a fix within an hour or two [23].
-

Sources

1. Battle Drones - a vibe coded RTS with audio-based multiplayer
2. X post by @steipete
3. X post by @steipete
4. X post by @steipete
5. X post by @voidzerodev
6. X post by @steipete
7. X post by @steipete
8. X post by @kentcdodds
9. X post by @kentcdodds
10. X post by @theo
11. Opus 5.5 is proof that we do need smarter models
12. patch the planet
13. X post by @GeoffreyHuntley
14. X post by @GeoffreyHuntley
15. X post by @GeoffreyHuntley
16. X post by @GeoffreyHuntley
17. X post by @OpenAIDevs
18. X post by @thsottiaux
19. X post by @thsottiaux
20. X post by @theo
21. AI for the Fortune 2000: How Tenex Builds an AI-Native SDLC — Alex Lieberman & CJ Hess
22. X post by @kr0der
23. X post by @kentcdodds