

Give the agent a goal and a check, then step back: Opus 5.5 prompting, hill-climbing and automated PR shepherds

Coding Agents Alpha Tracker

2026-10-29

Give the agent a goal and a check, then step back: Opus 5.5 prompting, hill-climbing and automated PR shepherds

By Coding Agents Alpha Tracker • October 29, 2026

Practitioners at Anthropic, OpenAI and T3 converge on the same approach: short goal prompts, checks the agent can run itself, and automation around review and merge. Also covered: Gemini 4 Carbon's limited launch, T3 Code's multi-machine orchestration and Simon Willison's game-building recipe.

Write less prompt and more verification

Addy Osmani says most people over-prompt Claude. For Opus 5.5, give one prompt that states the goal and how Claude should verify it reached that goal. Start with `/effort medium` and adjust from there [1]. If the goal is still fuzzy, ask Claude to interview you until it can write a spec, then start a fresh session to build from that spec [1]. Claude can run unattended when it has a pass/fail check or a number to beat, with reports it can read. For long background tasks, `/goal` has a separate model check a condition after every turn [1].

His other recommendations: - **Trim your instructions for each new model.** Over 80% of the Claude Code system prompt was removed for Opus 5 and Fable 5 with no measurable drop on coding evals. Cut your own instructions, add back only what the model misses, and use `/doctor prompt-audit` to flag instructions written for older models [1]. - **Subagents:** ask for them only for large standalone jobs such as audits or migrations. On small jobs they add cost and latency, and they don't make the agent smarter by default [1]. - **Cloud sessions** run in a VM with a fresh clone of your repo, so push your commits before starting one (`claude --cloud`). Parallel sessions only pay off once a

single session can finish without you. Projects, in beta on Pro and Max, track which threads are done, which PRs are ready and what needs your input [1]. - **Review:** on the Claude Code team, Claude reviews and fixes most changes at least once before a human looks. `/code-review` runs the review in a subagent that hasn't seen the conversation [1].

In a talk with Sonar, Osmani said Claude does the first-pass review and has to try the change the way a user would: browser checks, end-to-end tasks, linting. Hooks rerun on every edit [2]. Write your team's standards in `Claude.md/agents.md`. Humans then check the agent's evidence, such as videos or screenshots of the UI working [2]. The tests he relies on to merge automatically are end-to-end, property and visual-verification tests [2].

Theo's new video turns the same idea into a hill-climbing recipe. For TS Rust, he told the model to run real projects against a pinned TypeScript version and keep going until the outputs matched exactly. Once they matched, he told it: "Do not stop until you're two times faster", with existing tests still passing [3]. Some goals are hard to check. For a Skate clone, he had the agent screenshot its builds and send them to a panel of LLM judges for comparison with real footage. He calls this crude, and says it burned at least \$800 in tokens on the skybox [3]. His main point is to keep asking the agent what is blocking it. Moving builds to snapshotted VMs on Boat, a company he has invested in, cut gate time from up to 70 minutes to 353 seconds [3]. He warns against doing this at API prices [3].

Automate the PR "last mile"

Brent Traut's Codex setup: 1. Install a PR-babysitting skill that addresses feedback, fixes merge conflicts and CI failures, and alerts you only when it needs you. He points to the one in the openai/codex repo. 2. Create a "Outbound" section in the ChatGPT app. 3. Create a " PR Shepherd" Codex thread scheduled every 15 minutes. It runs the skill on each Outbound thread until the PR is green, merges it, then archives the thread [4].

After requesting review, he drops the thread into Outbound and walks away [4].

Gemini 4 Carbon: strong coding scores, limited access

Artificial Analysis reports that Carbon (high), running in Antigravity CLI, scores 59 on its Coding Agent Index. That is 3 points behind Claude Code with Opus 5.5 (max, 62). It ranks second on Terminal-Bench 4.0 (64%) and FrontierCode v1.1 (51%) [5]. It is only available to selected users so far, with a 50% launch discount and no confirmed end date [5]. Access starts with cyber defenders, and Logan Kilpatrick says Google is working to make it more widely available soon [6]. Kilpatrick says Carbon cut successful cheating on Terminal-Bench 4.0 to 0.0%, unsupported claims by about 2× versus Argon, and overreach by 26% [7]. Matthew Berman reports pricing of \$2/\$10 per mil-

lion input/output tokens during an introductory period of unspecified length, then \$4/\$20 [8].

T3 Code: one MCP server controls a fleet of machines

Theo showed T3 Code starting 8 threads on different machines through one MCP server, driven from Grok Bot [9]. It works over the internet, not through computer use, so it still runs with his laptop off [10]. To try it, enable it under Labs on the latest nightly. Theo calls it “VERY early” [11]. Also new: `cmd+k` → “triage” clones T3 Code’s source for your exact version, gives your agent a debugging playbook, and files an issue if the bug is T3 Code’s fault [12].

Building games that are fun

Simon Willison says Opus 5.5’s sense of fun beats GPT-6 Astra’s [13]. His recipe is to start in plain chat with **Build a mobile-friendly three.js game where...** and switch to Codex or Claude Code once the proof of concept works [13]. Two prompts did a lot of work. One prompt that included “Think hard about how to make sure it is both fun and beautiful” produced a game in one shot [13]. “Make it a LOT more fun” turned a dull 2D RTS into Battle Drones [13].

Memory and scheduled agents (LangChain)

- LangChain says over 70% of Managed Deep Agents runs come from schedules, not from someone typing a message [14]. Its blueprint is a cron server that takes a prompt plus the skills or permissions the run needs, adds random jitter so runs don’t fire all at once, and gives agents a controlled way to create and edit their own schedules [14].
- OpenWiki is an open-source CLI that builds a wiki of your codebase so coding agents don’t rediscover the architecture on every task. Each claim in the wiki is tied to a hash of its source lines, so a claim gets flagged as stale without a model call when those lines change [15]. LangChain then added an evidence graph and tested it on cross-document questions. It reports 27% higher accuracy, 2× citation consistency and 58% lower cost per correct answer [16].

Short notes

- **Kent C. Dodds** ties each Kody feature flag to metrics. A “Rerank” flag showed 7.6× more executions and a 5× lower error rate, so he will enable it. The same system led him to delete another feature [17, 18].
- **Riley Brown** uses a Claude Project as a coordinator. It sends work that needs his computer to local sessions and research and artifacts to cloud sessions, and he runs it from his phone [19]. He keeps 30 apps and 20 skills

in Composio so any agent platform can use them. Shared memory across platforms is still unsolved for him [19].

- **OpenAI:** eligible ChatGPT Pro and Business subscribers can claim up to \$500/month in API credits [20].
- **GitHub Octoverse:** merged PRs tripled in under a year to 220M per month. Willison reads this as the effect of agentic engineering [21].

Sources

1. X article by @addyosmani
2. How Anthropic Merges 30,000 Code Changes a Day - Manish Kapur, Sonar & Addy Osmani, Anthropic
3. How To Give Claude Unlimited Usage
4. X post by @btraut
5. X post by @ArtificialAnlys
6. X post by @OfficialLoganK
7. X post by @OfficialLoganK
8. Google “Carbon” just won everything...
9. X post by @theo
10. X post by @theo
11. X post by @theo
12. X post by @theo
13. The October inflection point for vibe game development
14. Why aren't you building proactive agents yet?
15. Building Evidence-Grounded Agent Memory with OpenWiki and Graphs
16. X post by @colifran__
17. X post by @kentcdodds
18. X post by @kentcdodds
19. I Spent \$10,000 Testing AI Agents (Here's the Stack I Actually Use)
20. X post by @OpenAIDevs
21. GitHub Octoverse 2026