Self-Critique Loops, Model Switching, and Opus 4.8 Signals

Coding Agents Alpha Tracker

2026-05-31

Self-Critique Loops, Model Switching, and Opus 4.8 Signals

By Coding Agents Alpha Tracker • May 31, 2026

Today's strongest signal is that better review scaffolding is extending what coding agents can do with higher confidence: critique-first prompting, autoreview loops, and explicit model switching are proving more useful than passively accepting the first answer. Also covered: Opus 4.8 benchmark signals, Claude Code reliability caveats, and a browser-based ASGI project worth studying.

TOP SIGNAL

- The best signal today is **critique-first prompting plus explicit review loops**. Lea Verou's Claude prompts start by accusing the code of being overengineered or architecturally incoherent, and steipete says the same trick works on Codex: when it says a review is "all good," tell it there is a bug and it keeps searching [1, 2]. steipete also says GPT 5.5 + `/goal` + `autoreview` + `crabbox` moved his prompts from ~30-60 minute tasks to often 4-10 hour tasks, with much higher confidence in the result [3].

TRY THIS

- **Force a second pass with specific critique language.** Lea Verou's copyable openers: "You overengineered this, there is a simpler way", "There is a smaller delta that buys us most of the benefits", "There is a more elegant way", and "This is not architecturally coherent" [1]. steipete's Codex version is even simpler: ask for bug review, and if it says things look fine, reply "there is a bug" and make it loop again [2].
- **Push longer jobs with `/goal` + `autoreview` + `crabbox`.** steipete says that exact GPT 5.5 stack turned ~30-60 minute prompts into 4-10 hour tasks and raised his confidence that the work was ready [3]. Start by

inspecting the two components he shared: Autoreview skill and crabbox [4].

"Yielding agents is a skill." [3]

- **Keep Codex running when one model taps out.** Jason Zhou says the official config lets Codex use third-party models like DeepSeek, Kimi, and GLM, and that open-source models can run through the codex harness [5, 6]. If you hit usage limits or just prefer another model, switch the backend instead of abandoning the workflow; his setup thread is the place to copy from [5].

WHAT SHIPPED

- **DeepSWE + Opus 4.8:** now live. On default high thinking effort, it scores 6% higher than Opus 4.7 xhigh while lowering average cost per task; Theo says that matches his experience [7, 8].
- **Claude Code field report:** Theo says he's seeing frequent random tool-call errors and write failures, and suspects Opus 4.8 may be better than Claude Code makes it look [9, 10].
- **Codex model switching surfaced:** the official config supports DeepSeek, Kimi, and GLM, while codex harness supports open-source models [5, 6]. Useful if you want one interface across multiple backends.
- **Emerging project worth studying:** Simon Willison says Claude Opus 4.8 produced a working approach for running Python ASGI apps in Pyodide with Service Workers, with a research PR, a basic demo, and a Datasette 1.0a31 demo [11].

GO DEEPER

- **Short demo clip — DeepSWE's Opus 4.8 result.** Watch the post. It's the quickest way to see the exact claim in context: default high thinking effort beats Opus 4.7 xhigh by 6% while lowering average cost per task [7].
- **Repo/spec to study — Autoreview.** Autoreview skill is one of the exact pieces steipete credits in his longer-running GPT 5.5 setup. If today's theme is review before trust, start here [3, 4].
- **Project to study — Pyodide ASGI in the browser.** Simon Willison's research PR and the two live demos are worth reading end-to-end because they show an agent producing a working system, not just isolated code snippets [11].
- **Tool worth inspecting — crabbox.** crabbox is the other named component in steipete's stack for pushing tasks beyond the usual short-agent horizon [3, 4].

Editorial take: today's edge is coming from better self-critique and model-switching workflows, not blind faith that the default agent pass is good enough.

[1, 2, 3, 5]

Sources

1. X post by @LeaVerou
2. X post by @steipete
3. X post by @steipete
4. X post by @steipete
5. X post by @aibuilderclub_
6. X post by @jasonzhou1993
7. X post by @datacurve
8. X post by @theo
9. X post by @theo
10. X post by @theo
11. Running Python ASGI apps in the browser via Pyodide + a service worker