Disposable Agent Code, Local DS4, and Open Models Passing the Swap Test

Coding Agents Alpha Tracker

2026-05-09

Disposable Agent Code, Local DS4, and Open Models Passing the Swap Test

By Coding Agents Alpha Tracker • May 9, 2026

The practical signal today is boundary-setting: use agents aggressively on cheap-to-regenerate code, keep context surfaces tight, and pay attention as local and open-model stacks become viable for serious coding workflows.

TOP SIGNAL

- **The highest-signal shift today: treat agent-written code as disposable scaffolding when change is cheap.** Mitchell Hashimoto says AI "slop" is useful because it enables fast parallel experimentation; he used agent loops in Ralph overnight to generate dozens of low-quality plugins so he could test a full GUI and plugin ecosystem, then regenerate them whenever the API changed because the cost of change was just tokens [1].
- **Kent C. Dodds is using the same boundary in practice.** His MCP-powered assistant Kody has produced 160k+ lines of code he has not read, which is acceptable for proving the idea works—but not for a finished product, where he says he would rewrite more intentionally from scratch [2]. Simon Willison's version is narrower but aligned: he already trusts Claude Code for routine production tasks, while warning that repeated success can create "normalization of deviance" and that real usage matters more than AI-generated tests/docs alone [3].

TRY THIS

- **Create a prototype-only lane for unstable surfaces.** Mitchell's pattern is concrete: keep core internals high-quality, but let agents generate the GUI, plugins, or provider layer while the API/SDK is still moving;

run loops overnight; regenerate on API changes; only ship that slop transparently to testers; then rewrite once the concept is proven [1, 2].

- **Ask for HTML, not Markdown, when you need an explanation you can inspect.** Thariq Shihipar's argument: HTML lets Claude produce SVG diagrams, interactive widgets, and in-page navigation for code explanations and reviews [4]. Simon Willison highlighted this exact PR-review prompt:

```
Help me review this PR by creating an HTML artifact that describes it. I'm not very familiar
```

[4]

Start with that pattern for gnarly diffs, then browse examples at thariqs.github.io/html-effectiveness [4].

- **Stop auto-loading every skill/tool into context.** Dillon Mulroy says he almost never wants skills auto-invoked, so he built custom tooling to toggle them on and off to keep them out of the context window unless needed; Armin Ronacher's terse alternative: prompt templates [5, 6]. Practical takeaway: keep reusable workflows off by default, enable them only for the current task, and use templates for repeat jobs.
- **For code search, start simple: agentic retrieval first, parallel fan-out second, embeddings later.** swyx says simple agentic RAG is good enough for many codebases—especially homogeneous ones—until you're dealing with something on the order of 10B-1T tokens; for search itself, fan out in parallel, e.g. four rounds of eight searches, instead of one-search-at-a-time contexting [7]. Then add semantic indexing for larger codebases, where tools like Cursor's embedding flow start to matter more [7].

WHAT SHIPPED

- **DS4 for DeepSeek Flash V4** — Salvatore Sanfilippo released DS4, an open-source inference engine for DeepSeek Flash V4 and his first major OSS project built primarily with AI-written code under human architectural control [8]. He says DeepSeek Flash V4 gives him a usable **1M-token** local context window on **198GB RAM** [8]. Benchmarks he cited: **~470 t/s prefill + 35-36 t/s generation on M3 Ultra**, **~250/25 on M3 Max** [8].
- **DS4's practical feature set looks agent-ready** — HTTP API server, streaming, tool streaming via PR, logging, tracing, and a disk-persisted KV cache treated as a first-class object [8]. The setup flow is unusually short: clone the repo, run `make`, download the model, start the server, then use it with pyagent / cloud code / open code; experienced pyagent collaborators told him that workflow felt like "product mode" rather than a toy [8].

- **Open-model swap test passed in production** — Caspar Brun says his org changed Fleet's internal model from **Sonnet 4.6** to **Kimi K2.6** and he "didn't even notice"; his claim is that open models are already good enough for most tasks outside the hardest coding work, at **5-10x lower cost** [9]. LangChain's framing: this is the year of open-source LLMs in agents [10].
- **Fleet added per-agent tracing control** — You can now enable or disable tracing at the individual agent level in Fleet, which Brace Sproul called a "big unlock" for getting full trace details only where you need them. Docs: langchain.com/langsmith/fleet [11, 12].
- **Codex migration got a direct path** — ChatGPT now exposes a switch-to-Codex flow; Tibo's practitioner summary was simple: "You can just migrate things" [13, 14].
- **Current model chatter from an agent lab** — swyx's coding-model shortlist right now: **Claude 4.6** and **GPT-5.3 Codex** [7].

GO DEEPER

- **10:18-12:34 — swyx on when simple agentic RAG is enough.** Good calibration if you're overbuilding retrieval: for many codebases, plain agentic search works fine until the corpus gets truly large or heterogeneous [7].

AI & Software Quality with Shawn Wang (aka swyx) (10:18)

- **15:44-16:45 — swyx on parallel search.** Watch this if your agent still does naive sequential retrieval; the useful bit is the fan-out pattern—multiple searches per round plus diversity across them [7].

AI & Software Quality with Shawn Wang (aka swyx) (15:43)

- **10:02-11:34 — antirez on the DS4 setup loop.** The benchmark numbers are nice, but the real hook is the workflow: clone, `make`, download model, run server, connect your coding agent stack [8].

DS4: ora su GitHub (10:02)

- **Study simonw/tools.** It contains the kind of narrow, useful artifacts that Claude Code is good at building quickly, including the Redis Array Playground and other small utilities [3].
- **Study inaturalist-clumper + simonw/inaturalist-clumps.** This is a clean end-to-end pattern worth copying: small Python CLI → git-scraped JSON → HTML frontend generated from a precise prompt against real data [3].

Editorial take: today's edge is boundary-setting—disposable prototype code, tighter context windows, and better search strategy beat blindly giving agents more rope [1, 5, 7].

Sources

1. X post by @mitchellh
2. X post by @kentcdodds
3. Vibe coding and agentic engineering are getting closer than I'd like
4. Using Claude Code: The Unreasonable Effectiveness of HTML
5. X post by @dillon_mulroy
6. X post by @mitsuhiko
7. AI & Software Quality with Shawn Wang (aka swyx)

8. DS4: ora su GitHub
9. X post by @caspar_br
10. X post by @LangChain
11. X post by @BraceSproul
12. X post by @LangChain
13. X post by @OpenAI
14. X post by @thsottiaux