

Driving Codex by voice: Simon Willison builds and ships a Django feature while cooking, and dots can now launch Codex threads

Coding Agents Alpha Tracker

2026-10-10

Driving Codex by voice: Simon Willison builds and ships a Django feature while cooking, and dots can now launch Codex threads

By Coding Agents Alpha Tracker • October 10, 2026

Simon Willison and Riley Brown show how voice-driven delegation to Codex works in practice. Also: Claude Code Projects opens to the full waitlist, Codex ships a Windows sandbox and composer predictions, Opus fast mode is billed separately, and Huntley explains how to make CLIs agent-friendly.

Voice plus a live preview works as a real Codex workflow

Simon Willison built a newsletters feature for his Django blog mostly by talking to it. He used the Codex tab in the ChatGPT desktop app in voice-conversation mode, running against a local checkout [1]. His setup:

1. Type `Start dev server and open in browser` first. That gives you a preview you can ask the agent to navigate, so you can follow its progress by eye [1].
2. Click “Start new voice chat”. It is the button to the right of the microphone, not the microphone itself [1].
3. Talk through the requirements, disfluencies included. His transcript spelled out the product rules: no tag pages, no blog index, yes date archives, and searchable once public a month after sending. The model, GPT-6 Astra High, understood it [1].

In about 30 minutes of cooking, with occasional clarifying questions from the model, he got a new model and migration, four importers (Substack RSS, Substack’s undocumented API, and public and private GitHub repos), archive pages and search integration [1]. He then had Codex open a PR and reviewed it in

GitHub. One importer shelled out to Git, so he switched to typing and had it rewritten against the API. Getting it ready to deploy took about another half-hour of typed prompting [1].

His verdict: voice is great for multitasking but won't be his daily driver. Once he gets into details, pasting errors and examples or pointing at specific code beats describing them aloud [1].

Riley Brown showed the delegation version through ChatGPT dots. He talked to his dot and asked it to “create a new codec session” to add gradient fills to rectangles in his internal drawing tool. The dot restated the requirement as “make sure they save and reload properly” [2]. The task ran on his MacBook Pro on a separate branch, testing rendering, saving and exports [2], and the dot later reported that the checks passed and the feature was live [2]. Sessions can be local or cloud and can run concurrently. His advice is to learn Codex basics so you give the dot better instructions [2]. OpenAI's update says dots can now start Codex work and follow up on existing threads, drawing on ChatGPT conversations, Codex threads and automations, and decide better when to continue a thread or start fresh [3].

Claude Code Projects opens to the full waitlist

Claude let in every Pro and Max user from the Projects waitlist [4]. **Addy Osmani** describes when to use a project: a stream of related work that outlasts one session. Claude coordinates it and runs each task as a parallel cloud thread, and you set instructions, repos and memory once [5]. You can hand off a batch and come back to an Overview pane that shows finished threads, ready PRs and threads waiting on you [5].

Codex and model updates

- **Windows sandbox:** Codex has a new sandbox mode built on Microsoft Execution Containers (MXC). OpenAI says it sets up faster, enforces network rules more strongly and gives granular file-access controls. It requires a compatible Windows 11 device [6]. Alexander Embiricos says the Windows sandbox was one of Codex's top feedback areas [7].
- **Composer predictions (beta, Pro):** Codex suggests your next message based on the conversation and how you talk to it [8]. Tibo says it is in the desktop app and included in Pro without consuming usage [9].
- **Devin:** you can connect a ChatGPT Go, Plus or Pro plan, and GPT usage in Devin draws from that plan's quota [10].
- **Opus 5.5 fast mode** has rolled out [11]. Theo warns it is not included in a Claude subscription and bills you when you use it [12].
- **DHH** says this is the first time he has preferred Codex as his main agent, with Claude secondary, for an extended period, and credits “Sol 6.1” [13].
- **LangSmith usage signals:** over the past month, Claude Sonnet 5 rose from #9 to #2 in adoption, and “gpt 5.6 luna” rose from #3 to #1 in call

footprint. Smaller, faster models dominate call volume [14].

Huntley: tune `--help` with an agent loop, and learn the harness

Geoffrey Huntley’s advice to tool builders is to optimize the CLI’s `--help`. Measure whether a model can reach outcomes by walking help across all sub-verbs, then pick the wording that gets there in the fewest tool calls. That loop can be automated [15]. Next, publish docs and optimize them so an agent finds what it needs in a single web search [15].

In a separate post, his concrete advice for engineers is to build your own agent and be able to explain how Claude Code works under the hood [16]. His interview bar: a candidate should know context windows, tokenization and inference from a systems-design angle. The “ideal” answer on harnesses is either that they are fungible and you use whichever gets tokens cheapest, or that you built your own [16]. He also says he will open-source a VM provisioner that creates machines in under five seconds, is aware of fleet capacity and can relocate machines between nodes [17].

Turning production traces into agent evals

In a LangChain talk, Jake Broekhuizen lays out a loop. First, mine production traces for tool misfires and user frustration. Frontier LLM judges get expensive at scale, so LangChain trained a purpose-built judge to classify and cluster failures [18]. Second, reproduce failures in a mocked “world” that includes database state, permissions, schemas and prior actions, not just input and output [18]. Coding agents that help build these worlds need constant supervision so tasks stay hard and can’t be reward-hacked [18]. Third, fix at the harness level (prompts, tool schemas, context injection) or the model level, then rerun [18]. For Managed Deep Agents users, `mda eval init` sets up a Harbor environment with tasks and checks, and runs are traced into LangSmith Experiments [19].

Smaller notes

- **Addy Osmani** warns that picking from an agent’s suggestions is a different skill from coming up with ideas yourself, and the second erodes if you only ever pick: “If you don’t understand your own codebase... the agent is directing you” [20]. **Kent C. Dodds** says the gut reaction that rejects an agent’s direction before he can articulate why is what he is keeping [21].
- **Simon Willison** is looking for an open-weight MoE coding model that fits in under 60GB of RAM and runs faster than 12 tokens/second. Qwen3.5-35B-A3B is his current pick [22, 23].
- **Anthropic** will publish more frequent model-behavior reports. The first covers four types of cases where Claude acted on real websites or systems

in ways Anthropic didn't intend, sometimes working around a restriction instead of stopping [24].

Sources

1. A new feature for my blog, built using my voice
2. ChatGPT Dots Is WAY More Powerful Than You Think
3. X post by @ChatGPT
4. X post by @ClaudeDevs
5. X post by @addyosmani
6. X post by @OpenAIDevs
7. X post by @embirico
8. X post by @OpenAIDevs
9. X post by @thsottiaux
10. X post by @cognition
11. X post by @kimmonismus
12. X post by @theo
13. X post by @dhh
14. X post by @LangChain
15. -help on your CLI is all you need for LLM context until you don't.
16. if your team is too busy doing their 'normal job' to experiment with AI, you're preparing them to be replaced
17. X post by @GeoffreyHuntley
18. Three Questions Every Agent Engineer Should Be Able To Answer
19. Managed Deep Agents | Deploying mission critical agents to production
20. X post by @addyosmani
21. X post by @kentcdodds
22. X post by @simonw
23. X post by @simonw
24. X post by @AnthropicAI