Codex Subagents Go GA as Specs and Review Become the Real Constraints

Coding Agents Alpha Tracker

2026-03-17

Codex Subagents Go GA as Specs and Review Become the Real Constraints

By Coding Agents Alpha Tracker • March 17, 2026

Codex subagents were the clearest release today, but the bigger pattern came from practitioners across tools: better specs, cleaner context boundaries, and tighter review loops are what actually unlock agent leverage. This brief covers the tools, workflows, clips, and projects worth stealing from right now.

TOP SIGNAL

OpenAI shipping **subagents in Codex** is the biggest practical release today: specialized workers let you keep the parent context clean, split work in parallel, and steer results as they come back. Simon Willison's follow-up makes the broader point—subagents are now GA in Codex, custom agents live in `~/.codex/agents/` as TOML, and the same interface pattern is already surfacing across Claude Code, Cursor, VS Code, and Gemini CLI. [1, 2]

> "a glimpse of a future where agents orchestrate agents" [3]

TOOLS & MODELS

- **Codex subagents / custom agents** — now GA after preview; default subagents include `explorer`, `worker`, and `default`, while custom agents can be defined in `~/.codex/agents/` and pinned to models like `gpt-5.3-codex-spark`. OpenAI's practical pitch is straightforward: cleaner parent context, parallel tasking, and live steering. [2, 1]
- **Fast-subagent tip for Codex Pro** — Alexander Embiricos says you can explicitly ask Codex to spawn subagents, and Pro users can use **Spark** for faster ones. [4]
- **Remote env setup is getting first-class support** — Claude Code now supports custom environments for remote runs via

`http://claude.ai/code`, desktop, and mobile, plus setup scripts for dependencies, settings, and configs before launch. Kent C. Dodds says Cursor agents already offer a full Linux VM with browser plus screenshot/demo-video support and custom startup setup. [5, 6, 7]

- **LangGraph Deploy CLI** — `langgraph deploy` is now the one-step path to LangSmith Deployment: the CLI builds a Docker image, provisions Postgres and Redis, fits CI/CD, and adds `list`, `logs`, and `delete` management commands. First-party templates now include `deep-agent-template` and `simple-agent-template`; quick start is `uvx --from langgraph-cli langgraph deploy`. [8]
- **openClaw is pushing more logic into plugins** — Steinberger says "everything can be a plugin now"; lots of code moved out of core, with faster performance and lower memory use overall, plus Claude/Codex/Cursor plugin bundle support. It still needs another day or two to stabilize. [9, 10]
- **Mistral Small 4** — new Apache 2 licensed **119B** MoE model with **6B active** parameters; Mistral positions it as one model spanning reasoning, multimodal, and Devstral-style agentic coding. It supports `reasoning_effort="none"` or `"high"`, and Simon Willison tested it via `llm-mistral`. [11]

WORKFLOWS & TRICKS

- **Subagent orchestration recipe**
  1. Define narrow specialists as custom agents in `~/.codex/agents/`. [2]
  2. Give each one a job, not a vague mandate — Simon's doc example uses `browser_debugger` to reproduce, `code_mapper` to trace the path, and `ui_fixer` to ship the smallest fix. [2]
  3. Keep the parent agent focused on coordination while workers handle exploration in parallel. [1, 2]
  4. Steer individual agents as evidence comes back instead of dumping everything into one growing thread. [1]
- **Spec pack before prompt**
  1. Spend **30-40%** of the task writing the spec: requirements, constraints, success criteria, stack, libraries, and UI components. [12]
  2. Put supporting docs in a `context` or `resources` directory. [12]
  3. Encode architecture and team best practices in markdown or via MCP so the model doesn't default to generic patterns. [12]
  4. State the **goal**, not just the task — Theo's chess-engine example failed because the agent inferred the wrong objective. [13]
- **Local-to-prod LangGraph loop**
  1. Install the CLI: `uv tool install langgraph-cli`. [14]
  2. Scaffold with `langgraph new` and pick the DeepAgent template if you want a fuller harness. [14]
  3. Set LangSmith and model-provider keys in `.env`. [14]
  4. Run `uv sync` and `langgraph dev` to test locally in LangSmith Studio

with traces and hot reload. [14]
5. Deploy with `langgraph deploy`, then manage with `logs`, `list`, and `delete`. [14]

- **Simon Willison's data-analysis pattern is reusable outside journalism**
  1. Work in **Python + SQLite**, optionally with Datasette. [15]
  2. Use agents for database Q&A, exploration, cleaning, visualization, and scraping — his workshop handout breaks the flow into those modules. [15]
  3. For UI work, serve a Datasette `viz/` folder and have Claude Code write interactive visualizations straight into it. [15]
  4. If you're onboarding a team, his workshop setup used GitHub Codespaces plus a budget-restricted Codex key; attendees consumed **$23** in tokens. [15]
- **Set a merge policy now**
  - Logan Kilpatrick's blunt read: the bottleneck has already shifted from generation to **code review**. [16]
  - Addy Osmani's rule of thumb: merge AI-generated changes when they're small/compartmentalized or backed by enough tests, and keep humans in the loop for harder maintenance work. [12]

PEOPLE TO WATCH

- **Simon Willison** — dropped two operator resources in one day: a NICAR workshop handout on using Claude Code and Codex for data work, and a fresh chapter explaining coding agents as `LLM + system prompt + tools in a loop`. Good if you want both hands-on workflow and mental model. [15, 17]
- **Addy Osmani** — best practical framing today on spec-driven development for agent workflows; useful because he pairs the spec advice with an explicit quality bar for merges and maintenance. [12]
- **Theo** — worth watching for showing both the upside of multi-agent orchestration on a large repo merge and the failure mode when an agent optimizes for the wrong implied goal. [13]
- **Logan Kilpatrick** — a short post, but probably the cleanest organizational warning of the day: your process is likely underprepared for AI-heavy review load. [16]
- **Kent C. Dodds** — credible firsthand signal on remote agents because he names the concrete features he actually uses in Cursor, and he discloses that he gets free usage rather than pretending it's a neutral review. [18, 7, 19]

WATCH & LISTEN

- **1:28-3:49 — LangGraph local iteration loop**: Best short demo today if you want to see how `langgraph dev` turns an agent into a local server,

surfaces traces in Studio, and hot-reloads prompt changes before deploy.

[14] 

Deploy CLI: The Easiest Way to Deploy Agents from Your Terminal (1:27)

- **25:09-26:15 — Theo on goal vs. task drift**: A very real failure case: the agent "succeeds" by satisfying the literal prompt while missing the intended goal. Useful calibration for anyone over-trusting long-running

agents. [13]

You don't want to be a manager. (25:09)

- **0:38-1:15** — **Addy's spec checklist**: Fastest clip in the batch for improving agent outputs tomorrow morning — constraints, success criteria, stack, libraries, and UI components, up front. [12]

Why You Need Spec-Driven Development in the Age of AI with Addy

Osmani and Tim O'Reilly (0:38)

PROJECTS & REPOS

- **deep-agent-template** — official first-party LangGraph starter for heavier agent workflows; adoption signal is that LangChain used it in the Deploy CLI walkthrough and paired it with one-command deployment. [8, 14]
- **simple-agent-template** — smaller starting point for the same `langgraph deploy` path. [8]
- **Trees heatmap gist** — concrete artifact from Simon Willison's workshop: Claude Code generated an interactive Leaflet.heat visualization inside a Datasette `viz/` folder over a large tree dataset. [15]
- **Cursor security agents** — not open source, but high-signal production usage: Cursor says it runs a fleet of security agents continuously on its own codebase and published automation templates for others. [20, 21]
- **openClaw plugin bundles** — watch this framework if you care about tool extensibility: Claude/Codex/Cursor bundle support plus a slimmer core means the project is moving toward a more modular agent surface. [9]

Editorial take: the stack is converging on the same playbook — write a better spec, fan work out to specialists, and spend the saved time on review instead of pretending raw generation is still the bottleneck. [12, 1, 16]

Sources

1. X post by @OpenAIDevs
2. Use subagents and custom agents in Codex
3. X post by @edbayes
4. X post by @embirico
5. X post by @_catwu
6. X post by @OmidMogasemi
7. X post by @kentcdodds
8. Introducing deploy cli
9. X post by @steipete
10. X post by @steipete
11. Introducing Mistral Small 4
12. Why You Need Spec-Driven Development in the Age of AI with Addy Osmani and Tim O'Reilly
13. You don't want to be a manager.
14. Deploy CLI: The Easiest Way to Deploy Agents from Your Terminal
15. Coding agents for data analysis
16. X post by @OfficialLoganK
17. How coding agents work

18. X post by @kentcdodds
19. X post by @kentcdodds
20. X post by @cursor_ai
21. X post by @cursor_ai