Codex Moves Up to Goal-Level Work While GPT-5.5 Field Reports Stack Up

Coding Agents Alpha Tracker

2026-05-26

Codex Moves Up to Goal-Level Work While GPT-5.5 Field Reports Stack Up

By Coding Agents Alpha Tracker • May 26, 2026

Today's sharpest shift is from one-shot codegen to long-running, test-backed goal execution. Inside: copyable Codex workflows, skill hygiene, Symfony orchestration, and the clearest GPT-5.5 practitioner signals.

TOP SIGNAL

- **The interface is moving from code generation to goal ownership.** Romain Huet says Codex can take a single ambitious `/slashgoal` and run on it for 3-5+ days — Peter Steinberger reportedly let one run for over a week — while the human mostly reviews diffs, comments inline, watches CI, and uses PR review as the control point. Simon Willison's red/green TDD advice is the matching guardrail: keep a test suite that stops agents from breaking old features while they make new changes. [1, 2]
- Huet also says more than 40% of tasks sent to Codex are non-coding automations, a sign that practitioners are already using these tools for broader work than code generation alone. [1]

TRY THIS

- **Run one big goal, then review at the diff/CI layer.** Give Codex a single ambitious objective — e.g. update a neglected legacy codebase and keep going until the goal is reached — and let it work uninterrupted. Then review inline diffs/comments, create the Git commit/push, monitor CI in the app, and turn on Codex Code Review for the PR. Keep a red/green test suite in place so the agent can add new behavior without silently breaking old features. [1, 2]

- **Use an image-first frontend loop.** Ask Codex to generate a design image, iterate on the static result first, then tell GPT-5.5 to implement it. Use the in-app browser to point at elements and give concrete edits like `change this font`, `make it larger`, or `move this to the right`. [1]
- **Trim skill bloat and trust tool-based recall.** Peter Steinberger's instruction when writing skills: tell the agent to `be token efficient` and `relax grammar`, because long skill descriptions get loaded into every context. Then run `skill-cleaner` to find the worst offenders, and let the agent recover forgotten details via tools instead of preloading huge slabs of context — Huet says GPT-5.5 can compact understanding and recall details later when needed. [3, 1]
- **Use read-only audits before cleanup.** KingBootoshi's Codex prompt was `do a FULL read only analysis on my Macbook to help me optimize storage`. Codex reportedly surfaced 500 GB of potential savings and a 116 GB `codex-tui.log` file; the replicable part is the first pass being explicitly read-only. [4]

WHAT SHIPPED

- **Symfony** — experimental spec + reference implementation for fleet-of-agents orchestration. Huet frames it as a world where you care about task completion, not staring at code: use Linear as source of truth and monitor which agents are executing or completing each task. [1]
- **Codex stack details** — Huet says the Codex CLI and harness are open source; the harness defines tools, environment, file access, internet access, and MCP servers, and is also part of post-training so the model already knows its tool setup. Adoption signal: he says Peter Steinberger's output doubled after switching to Codex, and he completed 40 open-source projects in one year. [1]
- **skill-cleaner** — Peter Steinberger released a skill to detect overly long skill descriptions. Repo: skill-cleaner SKILL.md. [3]
- **GPT-5.5 field signal from DHH.**

  > "All steering, no handwriting." [5]

  DHH says GPT-5.5 has produced more "I can't believe it's this good" coding moments than any model since Opus 4.5, and says it fixed Omarchy 4 Alpha's busted webcam selector in under 2 minutes. [5, 6]
- **Chorus/iMessage build flow** — Riley Brown shared a path to go to http://chorus.com, add Claude to iMessage, and ask it to build an iOS app; he says the method works with any agent. [7]

GO DEEPER

- **11:17-12:51 — Image-first frontend loop.** Best short clip today for anyone building UI with agents: static design first, live browser second, implementation third. [1]

AGI, le futur de l'ingénieur, Codex, - OpenAI head of Developer Experience x Hexa (11:16)

- **13:48-15:00 — Codex as a workbench, not just a code generator.** Inline diff review, Git, CI, PR review, plus the note that 40% of tasks are al-

ready non-coding. [1]
AGI, le futur de l'ingénieur, Codex, - OpenAI head of Developer Experience x Hexa (13:47)

- **18:04-19:15 — Why raw context-window talk is fading.** Huet's point: agentic compaction plus tool recall matter more than stuffing every-

thing into the prompt upfront. [1]
AGI, le futur de l'ingénieur, Codex, - OpenAI head of Developer Experience

x Hexa (18:04)

- **24:30-25:09 — Symfony orchestration pattern.** Very short clip, but the idea is durable: Linear as source of truth, multiple agents executing tasks, human supervising outcomes. [1]

AGI, le futur de l'ingénieur, Codex, - OpenAI head of Developer Experience x Hexa (22:34)

- **Read:** Simon Willison's red/green TDD for agent code — simple rule, high leverage. [2]
- **Read:** Peter Steinberger's skill-cleaner SKILL.md — good example of using a skill to police other skills. [3]

Editorial take: the durable edge today is ambitious async goals plus tight guardrails — tests, lean skills, and review/CI checkpoints — not longer prompts. [1, 2, 3]

Sources

1. AGI, le futur de l'ingénieur, Codex, - OpenAI head of Developer Experience x Hexa
2. X post by @simonw
3. X post by @steipete
4. X post by @KingBootoshi
5. X post by @dhh

6. X post by @dhh
7. X post by @rileybrown