Codebase Q&A First, Then Let the Agent Edit

Coding Agents Alpha Tracker

2026-06-29

Codebase Q&A First, Then Let the Agent Edit

By Coding Agents Alpha Tracker • June 29, 2026

Boris Cherny shares a production-tested Claude Code playbook: start with repo Q&A, pin context in `CLAUDE.md`, and force plan/verification loops before edits. Also: Claude Code's GitHub app and Hermes' self-healing, routed-agent stack.

TOP SIGNAL

Boris Cherny's strongest production lesson from inside Anthropic: use the coding agent as a **codebase interrogator first and an editor second** [1]. He says new technical hires start with code Q&A, then gate edits with `before you write code, make a plan and run it by me`; this is now part of an onboarding flow that he says went from 2-3 weeks to 2-3 days, and roughly 80% of Anthropic's technical staff use Claude Code daily [1]. Jon Udell's companion point, surfaced by Simon Willison, is the timeless framing: agentic development should stay *our* loop, with agents invited into a transparent human-led process rather than treated as a black box [2].

TRY THIS

- **Start with repo Q&A, not codegen (Boris Cherny).** Open Claude Code and ask questions like `how is this class instantiated?`, `why does this function have 15 arguments? look through git history`, or `what did I ship this week?`. Cherny says this is the default onboarding move at Anthropic; Claude explores the repo and git history without indexing, keeps code local, and does not train on your code [1].
- **Force plan → verify → iterate.** Use Cherny's exact opener: `before you write code, make a plan and run it by me` [1]. Then give the agent a way to check itself—unit tests, Puppeteer screenshots, or iOS emulator screenshots—and let it run 2-3 feedback rounds; for UI work, you can also drag in a mock image and have Claude implement against it [1]. If

it's clearly on track, `Shift+Tab` moves Claude Code into auto-accept edits mode, and `commit push pr` will usually handle branch, commit, push, and PR creation [1].

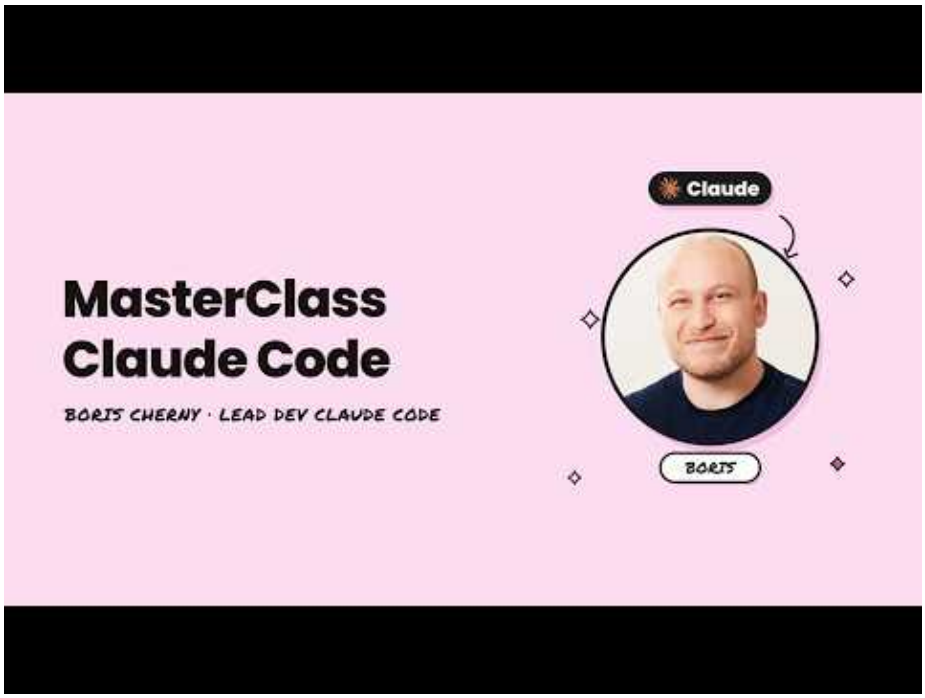
- **Externalize durable context into `CLAUDE.md`.** Put a short `CLAUDE.md` at the repo root and check it into git for the team; fill it with common bash/MCP commands, style guide rules, architectural decisions, and core files [1]. Add nested `CLAUDE.md` files for subdirectories, keep them short so they do not waste context, and use `#` in-session when you want Claude to remember a new rule and fold it into the file [1].
- **Use the agent like a UNIX utility in ops and CI.** Cherny's `claude -p` pattern: pass a prompt, allowed tools, and JSON or streaming JSON output, then pipe in things like `git status`, Sentry CLI output, or GCP logs [1]. For parallel work, run multiple sessions via tmux or SSH and isolate them with git worktrees instead of one giant session [1].

WHAT SHIPPED

- **Claude Code GitHub app.** Anthropic announced a GitHub app that lets you mention Claude directly on any GitHub issue or PR; Cherny also says Claude Code is already used daily by roughly 80% of Anthropic's technical staff, including researchers, which is a stronger adoption signal than most launch posts [1].
- **Hermes is pushing a more modular agent stack.** In Matthew Berman's tutorial, Hermes shows built-in coding skills plus three patterns worth stealing: self-healing when a skill hits an unseen error, siloed agent profiles instead of one bloated assistant, and per-task model routing for vision, compression, and web extraction [3]. In the demo, a Manim skill turned `make a cool video explaining how exponentials work` into a 58-second animated MP4 [3].

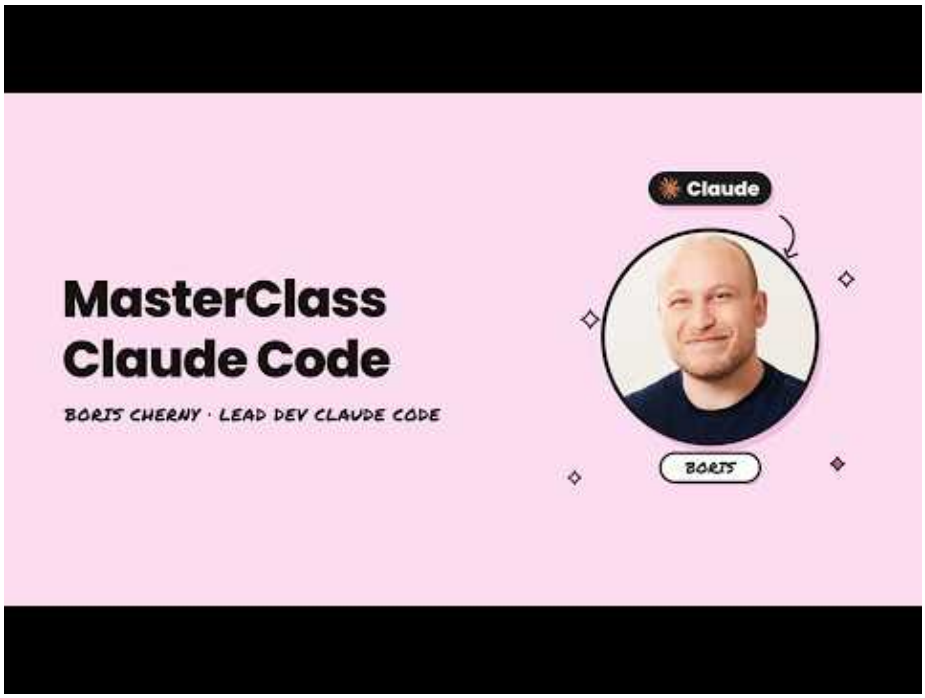
GO DEEPER

- **4:05-7:16 — Boris Cherny's Claude Code walkthrough on codebase Q&A.** Best clip if you're still using coding agents mainly for edits: Cherny shows the exact class-usage, git-history, and weekly-ship prompts that made this the first-day workflow for Anthropic onboarding [1].



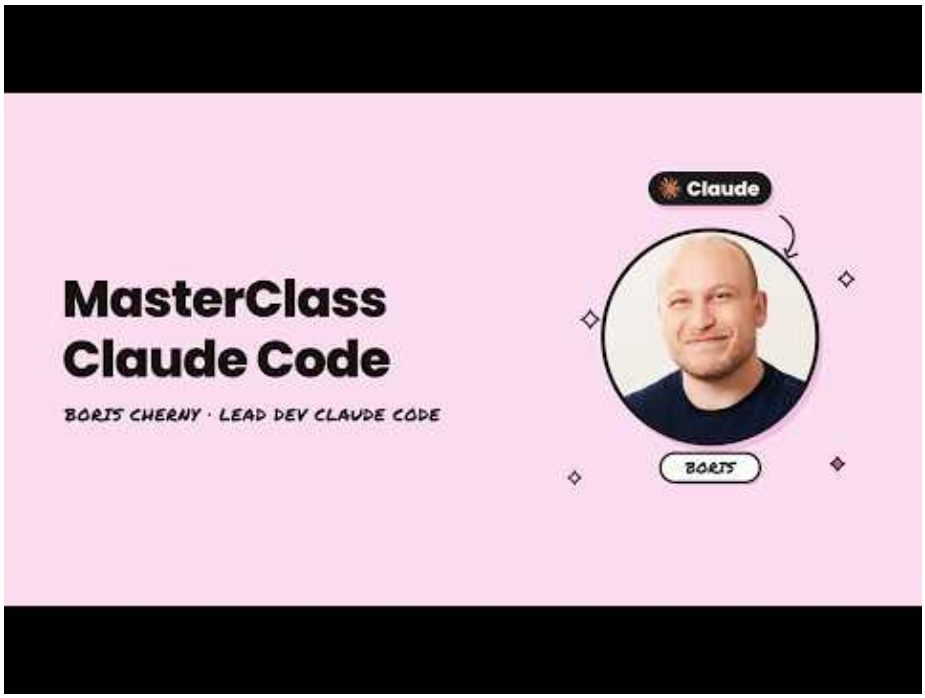
MasterClass du créateur de Claude Code : Boris Cherny (4:05)

- **12:25-13:59 — Cherny on `CLAUDE.md`.** Strong reminder that context engineering does not need to be fancy: short repo-level instructions, nested directory-level context, and team-shared conventions beat re-explaining the same codebase every session [1].



MasterClass du créateur de Claude Code : Boris Cherny (12:24)

- **21:08-23:07 — Cherny on `claude -p` for CI and incident response.** Worth watching if you want the agent outside the chat UI: JSON output, shell piping, and log triage make the CLI feel like a programmable UNIX tool [1].



MasterClass du créateur de Claude Code : Boris Cherny (21:08)

Editorial take: the highest-signal agent workflows still look human-owned—ask better questions, pin context to files, require a plan, and give the model a way to prove its work [1, 2].

Sources

1. MasterClass du créateur de Claude Code : Boris Cherny
2. Quoting Jon Udell
3. “The best thing since OpenClaw” (Hermes Tutorial)