Episode Book

EPISODE #6597

Why Terminal-First Developers Embraced AI Agents

PUBLISHED	14 November 2026 at 15:13 UTC
DURATION	20:07
LISTEN	myweirdprompts.com/episode/terminal-developers-agentic-ai-81473/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	omnivoice
PIPELINE	V5.26
SOURCES	25 cited by the research (listed at the end)
BOOK GENERATED	15 November 2026 at 11:16 UTC

SYNOPSIS

Developers who spent years avoiding graphical interfaces are now leading the adoption of agentic AI. Using tools like Void Linux, dwm, tmux, and git worktrees, terminal-first programmers are running two to four concurrent Claude Code agents like an aircraft carrier flight deck. This exploration covers the 1,300-app TUI ecosystem, why Bryan Lunduke lasted only ten days without a GUI, and how 1970s multiplexing became the ideal architecture for supervising AI.

TRANSCRIPT



Corn

Twenty years of Linux, Daniel says, and a love-hate relationship with the command line the whole time. The love part is new. He writes that Claude Code is the only CLI he's ever actually enjoyed, and that he's now learning other command-line tools just by watching an agent use them every day — which, he admits, is kind of cool.



Herman

That's the detail I keep turning over. The thing that made him hate CLIs for two decades was memorizing flags. Now the thing teaching him the flags is a program that lives in the CLI.



Corn

Thecure coming from inside the disease. His policy, though, stays sane: GUI where it makes sense, CLI where that makes more sense, never going out of his way. His question is about the people who do go out of their way — who they are, what their setups look like, and whether the agentic AI wave is now meeting that crowd head-on.



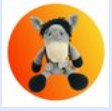
Herman

And that's a live question, because the answer is stranger than the question. Let's start with the setups, because the extreme edge is not one choice. It's a stack.



Corn

Layer cake.



Herman

Bottom layer, the distro: something like Void Linux, chosen precisely because it's minimal and doesn't hand you anything. Next layer, the window manager: dwm, or i3 — tiling window managers where the GUI hasn't been abolished, it's been demoted. You still have a screen, but it's just a canvas of terminal windows, and you move between them with keyboard chords lifted straight out of vi.



Corn

So the graphical interface exists to hold text boxes, and the text editor's keybindings run the house.



Herman

That's the ideological move, yes. And then the top layer — the people with no X server at all, just a console and tmux. There's an archetypal post on the unixporn side of Lemmy: Void Linux, dwm, terminal, statusbar, launcher, and every single line ends the same way. "My build." The statusbar is my build, the terminal is my build, the launcher is my build.



Corn

Because those tools ship as source.



Herman

Suckless tools, exactly — dwm, st, dmenu. You don't configure them with a settings file. The configuration is the C code. You patch it and compile it yourself, so "my build" isn't bragging, it's just literally what it is. Your window manager is a file you own.



Corn

My relationship with my window manager is that it's there when I turn the computer on. Anything past that feels like a marriage.



Herman

The scope of what you can replace is the part outsiders underestimate. There's an index of terminal apps — several, actually, that's itself a symptom — organized by the thing you'd assume needs a GUI. Email: Neomutt, aerc, matcha. There's even a command-line client for Hey.com, the email service whose whole pitch was a fancy graphical interface.



Corn

They built a text version of the product whose selling point was not being text.



Herman

News: Newsboat for RSS, circumflex for Hacker News, Browsh, an entire web browser rendered as text. Social: Zulip has an official terminal client, Bluesky works in the terminal, and then there's my favorite artifact of the whole subculture, itter.sh. It markets itself as social media for purists, and the network is an SSH endpoint. You register by SSH-ing to a register address, then you log in by SSH-ing to the site like it's a server from nineteen ninety-two.



Corn

A social network where the login screen is a terminal prompt. That's not a website, that's a bunker.



Herman

System administration goes the same way — btop, k9s for Kubernetes, lazydocker, yazi and lf for files, rainfrog for databases, a TUI for Swiss rail timetables, an expense tracker called Bagels. The big curated list on GitHub is over twenty-one thousand stars. A newer one claims thirteen hundred apps, synced weekly. That's the real answer to how far you can push it: far enough that nobody can hold the catalogue in their head.



Corn

Thirteen hundred terminal apps. Somewhere in there is a text-mode stock ticker and a man who needs it.



Hilbert

Hilbert's Anteater Trivia. Giant anteaters use their front claws to tear open termite mounds. I once tore open a memo with a letter opener and it took me the better part of an afternoon.



Corn

I'm not engaging with that.



Herman

So that's what they use — the harder question is why. And the reasons are more pragmatic than the aesthetic suggests. The best essay I've read on it, from a developer named Ian Johnson, makes four arguments. First, it's not stopwatch speed, it's latency between thought and action. His line: the five-click version of a task gives the brain five chances to lose the thread. The one-keystroke version gives it zero.



Corn

That one holds up. Five clicks is five decisions you didn't ask for, each one a chance to notice something shiny.



Herman

Second, composability. His phrasing is almost religious: pipes are not a feature, they're a worldview. The bar between two commands says whatever the first one wrote, the second one can read. No adapter, no plugin. You don't need a GitHub-for-Jira integration if one tool prints JSON and another reads it.



Corn

Third and fourth.



Herman

Third, resource efficiency — command-line tools spool up in an instant, do their thing, and leave. Fourth, and I think this is the quiet heavyweight: stability of the contract. His Makefile from a college C class still runs, except underneath it docker compose replaced the compiler. Different decade, different tool, same two words in his muscle memory. Every IDE button is a shell command underneath — the button is a convenience, the command is the contract.



Corn

That's the argument I'd actually defend. The button can be redesigned by somebody in California on a Tuesday. The command is mine until they change the kernel.



Herman

Now here's the correction, and it goes right at Daniel's framing. He calls them purists — everyone does, and I looked, and nobody in that world calls themselves that. No manifesto, no name for the movement. Johnson has a whole section titled "I'll use the GUI when it earns the slot." These people argue on latency and composability, not purity. Daniel's policy — GUI where it makes sense, CLI where that makes more sense — is the same policy with a different default setting.



Corn

Same religion, different pew.



Herman

Even the hardcore camp contains its own dissent. There's a thread from the suckless crowd where one poster says dwm needs a few too many patches to work how he likes, and another just replies, flat: bloat is unironically good. Emacs, GNOME, systemd, all good.



Corn

The realest thing anyone's said about an operating system in a decade.



Herman

And the empirical record on going all the way is brutal. Bryan Lunduke, professional Linux evangelist, attempted thirty days fully GUI-free for Linux Journal — no X server, just a terminal and him. He lasted ten days. Ten out of thirty, tapping out a third of the way in.



Corn

A man whose job was enthusiasm couldn't hold the position.



Herman

But then there's the counter-case, and the contrast is the whole point. A developer named borko, a few years back, switched the laptop he uses for writing to a fully command-line environment — no GUI at all — and adapted in a day or two. Found it great.



Corn

So the difference between collapse and conversion is scope.



Herman

Domain-specific extremism. The writing laptop is a terminal because writing is a text job. The machine with the photos still has a desktop. Almost nobody lives a hundred percent in the terminal — what looks like total abstinence from outside is usually a set of scoped decisions that add up. Corn, can I ask you something, because you're the one on this show who defends the slow, repairable, old option on principle. Does the ten-day collapse change your view, or does it just mean Lunduke chose the wrong battlefield?



Corn

It confirms my view, which is annoying for you. The old tool wins where it was already winning. Writing in a terminal makes sense because writing is keystrokes. He tried to do everything — browsing, media, the whole life — and discovered that some of life is pictures. The failure wasn't willpower, it was category error.



Herman

That's the cleanest statement of it I've heard. Which leads us to the second half of Daniel's question, because this is where the story stops being about nostalgia and gets strange. The terminal-first crowd didn't just adopt Claude Code. They rebuilt their entire workflow around it, and the terminal is the reason.



Corn

Explain that, because from outside it looks backwards. The people who avoided GUIs on principle embraced the newest, flashiest AI thing.



Herman

Jan Smrcka's argument is the one everyone converges on: GUI-based AI tools lock you into their editor. One agent, one context, one project at a time. In the terminal, everything is composable. Tmux gives each project its own session, git worktrees give each task its own copy of the code, and you can run several agents at once on the same repository without them stepping on each other.



Corn

Hold on. The pitch for the terminal was always one human, one keyboard, total focus. Now the pitch is one human supervising four software employees.



Herman

That's exactly the shift. One practitioner describes it as an aircraft carrier rather than a writing desk — a flight deck with several agents in the air, one refactoring an API, one fixing a flaky test, one writing migrations, and he's not editing in any of them. He's the air traffic controller, jumping between panes, reviewing diffs, sending the next instruction. Two to four concurrent agents, run through a small orchestrator script.



Corn

The man is not coding anymore. He's dispatching.



Herman

Clive Shirley, same architecture, put it in five words: the terminal is still the IDE. He's got a script where a ticket number becomes a worktree plus a tmux session in one command, and he dates the principle to twenty twenty-two — before the current agents were even good. The tools changed, the shape didn't.



Corn

What I find funny is why the terminal wins. Not the keyboard, not the purity. Multiplexing. Tmux and worktrees are nineteen-seventies machinery for sharing one machine, and it turned out that was the exact substrate for running a staff of agents. The GUI tools can't do it structurally.



Herman

The GUI was built around a single human looking at a single file. That's the sentence from the aircraft-carrier piece, and it's correct — the IDE's whole anatomy is wrong for this. The terminal was never designed for any one person's ergonomics, and that genericness is what aged into an advantage.



Corn

Forty years of being nobody's dedicated interface finally pays off.



Herman

And the tooling is following. The TUI ecosystem is now growing agent-shaped tools. There's a diff viewer called hunk, described as review-first, built for agentic coders — used by Mitchell Hashimoto, who is not a casual user. Lumen generates commits with AI from the CLI, oyui is a terminal merge tool. The catalogue Daniel asked about isn't static; it's mutating toward the agents.



Corn

Now the adoption side. Is this actually mass behavior or fifty bloggers?



Herman

It's mass. Stack Overflow's survey this year, twelve thousand respondents: Claude Code at sixty-five and a half percent, ahead of GitHub Copilot. JetBrains has workplace use at thirty-nine percent, up from eighteen percent in January. Pragmatic Engineer's survey, among people who use agents regularly, seventy-one percent. The CLI agent is the most-used agent experience in the industry, full stop.



Corn

Sixty-five percent. The terminal went from hazing ritual to onboarding standard in eighteen months.



Herman

One honest caveat — and I want to state it plainly — is that nobody's survey segments by terminal preference. The claim that the terminal-first crowd adopted agents fastest rests on the sheer volume of practitioner write-ups, not on a measured comparison. What we can say is the write-ups exist in unusual density and they all describe the same architecture, which suggests real convergence rather than a few loud individuals.



Corn

That's fair. The pattern is overdocumented but unmeasured, which is the most developer thing possible.



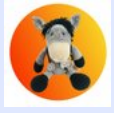
Herman

There's one more piece to Daniel's own observation, and I think it's the subtlest thing in the prompt. He says he's learning CLIs by watching an agent use them daily. Think about what that actually is. The agent's tool calls scroll past as it works — the command, the flags, the output, in context, for a real task. It's a live, contextual man page that performs the task for you while it demonstrates it.



Corn

The documentation writes itself as a side effect of the work.



Herman

The thing that made CLIs hard — memorizing flags nobody uses often enough to keep — is being solved by a program that lives inside the CLI. That's why the twenty-year haters are converting, and I'd bet it's also why Daniel, a perfectly normal Ubuntu user with a sensible hybrid policy, is suddenly enjoying the terminal. The barrier wasn't the interface, it was the tuition.



Corn

Nobody wanted to pay four years of flag memorization up front. Watching an agent work is learning by apprenticeship, except the master never sleeps and doesn't mind repetition.



Herman

Though — and this is where it gets contested — not everyone thinks the terminal should be the destination. There's a YC company from this year, Omnara, whose entire product is pulling Claude Code sessions out of the terminal onto web and mobile. Their pitch: it's like Claude Code Desktop, except you can continue your sessions on your phone.



Corn

So one movement is digging deeper into the terminal, and a funded company is building the exit.



Herman

The meeting Daniel asked about is real, and it's producing both. The terminal-native people are deepening — more agents, more panes, the air traffic controller life. And a whole other crowd wants the agents reachable from the sofa. Whether those are two audiences or the same audience at different hours, I don't know.



Corn

It's the same audience. Nobody who runs four agents in tmux does it for pleasure on Sunday. The phone front-end is the off-duty shift of the same job.



Hilbert

Purist. That's the word I'd quarrel with.



Hilbert

I ran a parts counter in the early nineties, Herman, before your GUIs were anywhere. Whole depot on green screens. Nobody was a purist. There was one door and everyone went through it. We called the system the counter, because that's what it replaced.



Herman

What did it run?



Hilbert

Serial terminals into a minicomputer in the back. Stock, invoices, ordering. There was a customer, Mr. Fielding, drove in from Ashdod way every fortnight, wanted his invoice numbers read out over the phone before he'd get back on the road. The system had function keys for all of that. F seven brought up a customer. I never used them. I typed the three-letter codes by hand for four years because I didn't trust keys that did things you couldn't see.



Corn

You distrusted the function keys on principle.



Hilbert

I distrusted any key that could be remapped without telling me. The codes couldn't be remapped. The old VT-two-twenty we did the ordering on is in my brother-in-law's garage in Rehovot now, still on its trolley. This itter thing you described. Does a man need to register to look, or can he log in quiet?



Herman

You have to register first, over SSH, with a username. And SSH sessions are logged on their end, same as any login.



Hilbert

Logged. Fine.



Corn

Four years of typing codes by hand to avoid one function key.



Hilbert

The codes never once brought up the wrong customer, Corn. I notice you two took forty minutes to say what I could've said at the counter in between phone calls.



Corn

The open question, then, is which way the meeting bends. Does the terminal-native crowd keep deepening — the aircraft carrier, four agents on the deck — or does the escape hatch win, and the agents drift out onto phones and browser tabs where the flags were never the point?



Herman

The Lunduke numbers suggest the deep end stays niche. Ten days of total abstinence for a professional enthusiast; a day or two when it's scoped to a text job. The terminal holds the work that is already text — and agent supervision, diff review, dispatch, is text work. That's the uncomfortable fit. It may hold on to the agents precisely because they made the terminal a management interface rather than a memorization gauntlet.



Corn

And the purist label was always ours, not theirs. Nobody in that world signed a manifesto — they set a default and argued about latency. What's changed is that the agents are rewriting the default for everyone else, Daniel included, from the inside. The question isn't who's pure anymore. It's what the terminal is for now that it's where the agents live.



Herman

If you've enjoyed this one, a review wherever you get the show helps other people find it, and the archive is all at my weird prompts dot com.



Corn

Our producer is Hilbert Flumingtop, who has never once trusted a function key. If you enjoyed this one, try episode seventeen, Cloud Render Superpowers; episode fifteen, AI Gets Personal; and episode fourteen, AGI's Crossroads. This has been My Weird Prompts. See you soon.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **GitHub, 21,094, created 2019-03-31, last push 2026-11-03; retrieved 2026-11-14. [primary]**
   https://github.com/rothgar/awesome-tuis

2. **GitHub, 1,300+ curated terminal user interfaces, created 2026-05-16, last push 2026-11-09; retrieved 2026-11-14. [primary]**
   https://github.com/alvinunreal/awesometui

3. **Stack Overflow Developer Survey 2026, n=12,255; retrieved 2026-11-14. [primary]**
   https://survey.stackoverflow.co/2026/ai/data/ai-code-agent

4. **JetBrains, AI Coding Agents: Adoption Trends, Developer Ecosystem Survey 2026; retrieved 2026-11-14. [primary]**
   https://blog.jetbrains.com/research/2026/08/ai-coding-agent-adoption-2026/

5. **ssp.sh, TUIs index; retrieved 2026-11-14.**
   https://www.ssp.sh/brain/tuis/

6. **GitHub, created 2026-04-07, because terminal UIs are having a renaissance; retrieved 2026-11-14.**
   https://github.com/dinakars777/awesome-tui

7. **Ian Johnson, Why I Still Live in the Terminal, published 2026-06-21; retrieved 2026-11-14.**
   https://blog.tacoda.dev/why-i-still-live-in-the-terminal-0d27918d85bb

8. **Bryan Lunduke, Without a GUI**
   How to Live Entirely in a Terminal, Linux Journal #300, 2019 (full text fetch failed; content via search extract); retrieved 2026-11-14.
   https://linuxjournal.rubdos.be/ljarchive/LJ/300/12805.html

9. **Abuse of Notation, Mastodon post, 2020-11-23; retrieved 2026-11-14.**
   https://mastodon.social/@borko/105260918916352621

10. **r/unixporn [dwm] Dream of the Endless, Void Linux setup, ~3 years old; retrieved 2026-11-14.**
    https://lemmy.ml/post/2173223

11. **4chan /g/ archive thread, 2020-06-30; retrieved 2026-11-14.**
    https://4archive.org/board/g/thread/76575912/discuss-suckless-utilities

12. **(window_manager)**
    Wikipedia, last edited 2026-08-26; retrieved 2026-11-14.
    https://en.wikipedia.org/wiki/I3

13. **Wikipedia, last edited 2026-10-27; retrieved 2026-11-14.**
    https://en.wikipedia.org/wiki/Tiling_window_manager

14. **Wikipedia, last edited 2026-09-18; retrieved 2026-11-14.**
    https://en.wikipedia.org/wiki/Text-based_user_interface

15. **Jan Smrcka, My CLI-First AI Development Workflow; retrieved 2026-11-14.**
    https://www.jansmrcka.dev/blog/cli-first-workflow

16. **Clive Shirley, How I Work Today: Terminal, Agents and a Second Brain, 2026-09-30; retrieved 2026-11-14.**
https://www.shirleyconsulting.co.uk/development/2026/09/30/how-i-work-today.html

17. **My setup for agentic-native engineering workflow with terminal environment; retrieved 2026-11-14.**
https://blog.duy.dev/my-setup-for-agentic-native-engineering-workflow-with-terminal-environment/

18. **Evgenii Pendragon; retrieved 2026-11-14.**
https://evgeniipendragon.com/posts/neovim-tmux-and-claude-for-fully-terminal-based-ai-driven-development/

19. **daily.dev, How I Rebuilt My Development Workstation for Agentic Work; retrieved 2026-11-14.**
https://daily.dev/posts/how-i-rebuilt-my-development-workstation-for-agentic-work-ubwwtf4gd

20. **JetBrains AI Pulse, January 2026; retrieved 2026-11-14.**
https://blog.jetbrains.com/research/2026/04/which-ai-coding-tools-do-developers-actually-use-at-work/

21. **The Pragmatic Engineer, AI Tooling for Software Engineers in 2026, 2026-03-03; retrieved 2026-11-14.**
https://newsletter.pragmaticengineer.com/p/ai-tooling-2026

22. **Nikolas Burk, State of Agentic Coders (Q1 2026), 199 responses; retrieved 2026-11-14.**
https://nikolasburk.com/blog/state-of-agentic-coders-results/

23. **Launch HN: Omnara (YC S25), 2026-02-12, 147 points, 161 comments; retrieved 2026-11-14.**
https://news.ycombinator.com/item?id=46991591

24. **Ask HN: What is your dev set up like?, 2026-04-02; retrieved 2026-11-14.**
https://news.ycombinator.com/item?id=47616250

25. **Why I Love the Command Line; retrieved 2026-11-14.**
https://rldane.space/why-i-love-the-command-line.html