Conductor CEO Charlie Holtz Walks Us Through His AI Coding Setup

16 MIN · YOUTUBE · HTTPS://YOUTU.BE/FQMLML9LAY4?SI=MCU_VTSYNE_VEJQO
https://youtu.be/fQmlML9Lay4?si=mcU_VTSYne_VEJqo

SUMMARY

Charlie, co-founder of Conductor, discusses the innovative features of their app designed to manage coding agents on Mac. He emphasizes the importance of voice interaction, experimentation, and the balance between AI and human input in software development.

- *Conductor allows users to orchestrate coding tasks using voice commands, enhancing productivity in open office environments.*
- *The app features a "caveman mode" for manual coding, but most tasks are handled by AI, which can be directed through voice.*
- *Experimentation is key; users can create multiple workspaces for testing ideas, with only successful ones being merged into the codebase.*
- *A new dashboard feature provides an overview of all agents' activities, promoting a CEO-like management experience.*
- *The app is built primarily in TypeScript, with a focus on maintaining clear boundaries between AI-generated and human-written code.*
- *Charlie advocates for a crafted user experience, emphasizing the importance of thoughtful UI design over AI-driven decisions.*
- *The team is exploring collaborative features for human-AI interactions, aiming to redefine the coding process as more of a conductor-orchestra relationship.*
- *Future developments may allow for more customizable workflows, akin to modding in video games, while maintaining a cohesive app structure.*

Hello, I am Charlie, the co-founder of Conductor, which is an app that lets you orchestrate a bunch of coding agents on your Mac. And we were YC Summer 24. Uh, and I'd love to show you my setup. So, a recent thing that I can't live without is this uh gooseeneck microphone. $20 on Amazon. we are all trying to talk to our computers more.

Um, one issue with having like an open floor plan office is that can be pretty distracting. So, one advantage of these is you can like lean over and whisper into Claude and be like, "Please uh merge PR 3475 and it's a little bit less disruptive. I we we all got these in an attempt to encourage more talking to computers." I spend most of my day in conductor. We're using conductor to build conductor. One thing that I do is I'm constantly kicking off new tasks. So I'm constantly um going commandn. Uh that was actually a sneak a sneak peek of something we are working on which is cloud workspaces.

But I'll I'll do command n and then I'll speak into my computer and say um can you take a look at the latest linear issue and give me a rough pass at how you'd solve it? Stuff like that. And then press enter. And then I can see that it's running in the sidebar. And while cloud is working I'll go to another chat. I'm very into keyboard shortcuts. to like try and make everything have a keyboard shortcut. So in this case I'll do command shift Y. I can see here that this workspace is ready to merge. So I'll take a look at it. Give Claude a quick review. Um in this case it's a pretty small PR and so it looks good to me. But quite often Claude won't get things exactly right and I'll give things a a comment like a GitHub style comment say uh this looks a little bit weird to me. Why do we need this? Press enter. get cloud running and then go back to a different workspace. A big part of um how I use conductor is like experimentation. Like I'm always kicking off workspaces to try different ideas. Most of them don't make it in. So like you can see we have like four PRs here that are in review, but like there's a bunch of random ideas that I've tried here that are in progress that you know may never see the light of day. If I like it, then it might get promoted to like an internal setting and then an experimental setting. Okay.

Okay. So, something I'm very excited about is on the go. Uh, I'm going to just speak into my phone and say, let's add a new feature where I can change the theme to hacker mode. And then I'm going to click conduct. And then my computer starts working on it and I can uh conduct on the go. >> You still write code today? >> No. Yeah. No. Very occasionally I will like edit Tailwind classes or like open up an IDE to change like a ENV file. We actually added a mode that we call caveman mode which is uh you click this and you can actually type with your keyboard and like make changes in a file. Once in a while you do need to like make a change to a file by hand, but it's like it's called caveman mode for a reason. Most of the time if I want small edits, I'll like highlight and then tell the AI um about my comments or I'll just like speak into my computer and say that button looks a little too wide like can you uh can you make it smaller? By the way, this thing is now ready to merge. Um so I just wanted to show you I can now click archive and it's gone from my side panel and like merge into the codebase and uh this one I can see that there are checks running and once it's finished I can just like click merge and and get it in. We recently added this thing called status in the left. So when something like is kicked off, it's in progress and then once there's a PR created, it's in review and then once it's merged, it goes into the done folder. We have this new concept of a dashboard page where from like one place you can see what all your agents are working on and then like take them to the next action. But we're still messing around with like what the what the interface should look and feel like. But the ideal is like you should feel like the CEO of a little company and you can see all your agents working for you and they'll bring you up like digestible reports and then you can point them in the right direction if they like need some correction or just merge it in if it looks good.

>> What are your other main applications, main software that you use? >> I use Telegram a decent amount to talk to my open claw. That's that's been a recent addition for me. I use spokenly for text to speech. That's what comes up when I press uh control space. It's actually running a local model. It's running parakeet. Um, I have a really beefed up computer, so it's like 128 gigabytes of RAM. Partly so I can like run local models like Parakeet. But as a side note, I have just recently ordered the MacBook Neo, like the the bottom of the line, lowest RAM, lowest memory. I I I got it basically to like force myself to like use the lowest spec option.

>> Are there any tweaks that you do still stand by that are the customizations that actually do matter? >> Uh, a couple things. Like we put a lot of time into our skills files and our like cloud MD. If I like open it up like you can you can see like this is uh probably a few hundred lines. There's like some interesting things in here like it's

we say engineering practices. We're a startup. You're probably used to writing enterprise code but that's not how we do things around here. And we have like a lot of things like that that we've like put into our cloud MD and our skills files over time. What else do I do? I always use fast mode. That's not a default. if you're trying to token max like you have to be in fast mode. I do use the context 7 MCP. I think that's pretty helpful to get documentation but other than that like use most of the most of the things out of the box. One core thing is that we always run claude and dangerously accept all permissions.

Um like that is not the default and that is uh the default way to run uh cloud in conductor. I think something that's really important to us is having like clear boundaries between uh well we call them slot free zones um and having like parts of the code base or like parts of the documentation that we like know is written by a human. It's possible like that the a the AI can contribute to the the slot free zones but like it has to be like every line has to be read by a human. I think this actually like served us pretty well. Um because if if you're not careful like the AI can like get in a vicious a vicious cycle where like it sees bad code and then it writes more bad code as a result. And the same thing can happen in like the positive uh positive direction. We have like some lines in our codebase that are like do not touch if you are an AI like this is for human eyes only.

>> What's the connector text stack? >> It's a towery app. So it's using the native uh Safari web renderer and the back end is technically Rust but we write almost everything in TypeScript. Um so it's like probably 90 95% TypeScript on the like the desktop app. The web app is Elixir. It's a it's a Phoenix app. It's a very small app because literally all you can do in it right now is just like log in. But I am I'm a huge Elixir fan and I am always like pushing for more Elixir in our codebase when we can. But um most of what we're doing is in Typescript.

Another thing we talk about is like don't let the AI be your architect. Even the concept of like a workspace here in the sidebar which in some ways is just like an abstraction around a work tree at least for right now. Like that's actually going to change soon. But even that like concept of a workspace like we as a human had to like think that through. The other thing is like like design and like interface decisions.

this concept of having like all your chats here on the left and then the chat in the middle and then the right sidebar where you can like review code changes or run your app like we put a lot of thought into into those decisions. And I think if you let the AI make your like UI choices for you, you can end up with something that like it just doesn't feel like crafted. It's like really important to us that it it feels crafted like even this decision. So like we we thought for a long time about how this open in button should work which is kind of funny because now there's like so many apps have like have the same pattern.

The thing that we were really thinking about is whether we should show the icons in the top. I was pretty against showing icons here at first because it just feels like okay in the top bar of our app like we're like advertising a different app but now I I really like it and it's like a clear visual of like what's going to happen um when you click it. I think something we would do a bit differently is building the core of the app um around like human written APIs and like contracts that the AI wouldn't contribute to as much. And then I think that like it's important to have a big chunks of your codebase be like have like free reign for the AI where you can just like throw a ton of different ideas at it and know that it's like not going to affect the core infrastructure. And I think

right now the the boundaries are a little murky and like um that's something we're we're working on improving. I think it's really important to us that we stay like a little ahead of the frontier, like push push people's comfort zones a little bit more than u they'd expect. When we first launched Conductor, uh most of the feedback we got was like this is crazy. Like I barely can manage like one cloud code or like one codeex. Like how am I going to manage like three or like even five? We also purposely made it so like you can't edit files directly. like we like made it so that like anytime a workspace like has to be a work tree and it has to then create a PR and then you have to merge it. So we really like enforced our workflow. I think like what's what's exciting but also hard about where we're at is like we have to constantly adapt to where like the models are going. So that's one reason like we are putting so much work into like cloud right now is right now like you shut your laptop and the agents are going to stop running.

But like feels like we're very quickly moving to a world where the agents are going to run for 10 times longer and they're going to be 10 times smarter and they're going to need to run in an environment that like isn't constrained by like your max like CPU. >> It seems like you're building conductor in a very opinionated way. How do you build a conviction behind your decisions? That that's a great that's a great question because yeah like it especially for like our audience they want a lot of like configuration and I do think it is important for the tool to like be flexible and to like feel like yours but the way we build conviction is we force ourselves to use it because actually we don't even force like we we just use it every day and so if it doesn't feel right like we like quickly can can decide but we we we're not big on analytics or like looking at like our AB testing or like It's very much a like gut feel. This feels right. Like when I click this, it feels right that it opens in the center. And that way I don't need a separate composer and I can type messages here and it all feels unified.

>> You sound like you default to cloud code in a lot of places, but conductor supports codeex too. When do you reach for codeex? >> Okay. I've recently actually been using codeex more. Codex is like the workhorse. it will power through like a specific problem or like uh it's not afraid to do a ton of tool calls and like debug something with me for a long time. Cloud I'll reach for when I want a little more like back and forth. I feel like Opus is just like a little more creative, like a little more uh of a partner. And so I would say like when I'm building out a new a new feature like I I probably would like instinctively reach for Opus. And then when I'm like okay now we just want to get stuff done like I I'll go to Codeex.

Why isn't just a terminal good enough? >> There's a reason uh we moved from terminal interfaces to like gooey interfaces in the 80s. Like I think humans are spatial visual creatures and like having a a command line interface just like feels like it's feels very like restrictive and I think it maybe works for the AI brains better than the human brains. But I think just like I want to know that okay my chats are over here and my like review panel is here. I can talk to the AI in the middle. I just think like yeah bottom line like humans are like visual visual creatures. I also think like like out of like a like zooming in a little bit like there's a lot that you can't do in a terminal um like that you can do with a uh user interface.

>> Let's talk about token maxing. >> Yeah. >> What's your high water mark on lines of code in a day or spend in a month? >> I think the highest spend was when we were starting out conductor like in July 2025. I spent $22,000 on tokens that month. Granted, that was with like previous generation of models. Um, and the lines of

code was must have been like tens of thousands that month. I'm very big on spending like on token maxing like using fast mode like think extra hard all like high effort all the time, but we're not big on lines of code. We uh we try and keep the lines of code minimal actually. There's a bunch of reasons for this, but I think like you can quickly spiral. Your codebase can spiral out of control if you're like not careful about the lines of code added. But I I I think about it very differently if I'm like starting up an app versus like working established codebase like Conductor.

>> What's different about your workflows today from say 6 months ago? >> On a lot of like hard PRs, I would open an IDE and make changes by hand. And I also use GitHub like the web app a lot a lot less now because I can just like review the code changes here in conductor and like add comments here if I need to. We do have like a lot of PR checks that run. Um and uh so that's why we recently added this like uh this checks tab which lets us just like add comments from GitHub like into Conductor.

>> What's the most surprising thing you've seen someone else do with Conductor? One was like someone built like a mobile version of Conductor by like hacking together a bunch of our I don't actually even really know how it works, but I know it's like spoofing like IPC calls to our desktop app, which is pretty interesting. I think honestly Gary has shown us a lot of what you can do with conductor. He is really putting it to the test. I think I've learned from him a bit about like how hard you can go on skills. like skills are very much like a first class thing in in GStack and it's like it's there's some like interesting ideas there I think like especially around like onboarding and we've added actually a specific mode for him called Gary mode which by default does not collapse any of the tool calls so you can see all the tool calls are default on collapse and you can even actually see uh Gary's face here if you're in Gary's mode >> what feels obvious to you and your team that the rest of the world doesn't fully understand yet >> like I think there's like a lot of cool stuff to explore with like collaboration between humans and the AIs. Should you be able to communicate with sub agents?

Should you be able to have like multiplayer chats where like multiple people are working on the same thing with the AIS? And then of course like the a metaphor we we'll often talk about is like feeling like the conductor of an orchestra. You like uh wave the baton and like the instruments are playing in unison and then once in a while you want to go to like the the trumpet player and be like, "Okay, you're out of tune." And then you want to like zoom out to like the string section and like uh you should play a bit faster, but then most of the time you're like conducting at the the orchestra level. Code is almost like uh sawdust now in that like it used to be that code was the thing you were building. It was like the structure. You were putting time into like in in into like crafting the code and now you're putting time into describing what you want and how you want it to be built.

And the code is almost just like sawdust that comes out of that process. And like that leads to like a lot of like interesting conclusions. Like one of them is like really what matters is your prompts. And like when the next generation of models come out, you can just like rerun your prompts again and then you'll get new code and the old code didn't really matter. I think that's one thing that like the world is slowly waking up to. I think like the submit a prompt like the prompt request feature is sort of like an early experiment with malleable software. I the the metaphor that I always think of when I think of malleable software is like video games and how like when

you play like Call of Duty like the structure of the game is the same for everyone and like the skeleton is the same but each person can like I don't know like use custom skins or like faster like reload speeds or whatever and like the same way you can like mod a video game. I want you to be able to mod Conductor and like yeah build in your own workflows a little bit. It's important that like the structure feels the same and like people want software that's like been crafted and been like really thought through. But I also, you know, like video game mods make make the game feel more like your own. And um I think that's going to happen with software as well.