How Claude Code’s lead designer builds with AI

12 MIN · YOUTUBE · HTTPS://YOUTU.BE/HKEDFUPBA4U?SI=BXL9VNIZY8O6QEID
https://youtu.be/hKeDfupbA4U?si=bXL9VNizy8O6qEiD

SUMMARY

The speaker presents practical workflows for using cloud code effectively, emphasizing the importance of optimizing work processes in a fast-paced environment. They demonstrate various tips and tools, particularly focusing on the use of work trees, automation, and leveraging AI tools like Claude to enhance productivity and streamline coding tasks.

- *Start work in a work tree to avoid conflicts when running multiple cloud instances.*
- *Use Claude for automating tasks beyond coding, including design suggestions and feature implementation.*
- *Implement a routine to check for front-end changes and ensure designer involvement in features.*
- *Utilize auto mode and fast mode for smoother workflows and quicker task completion.*
- *Regularly use Claude to verify features and automate PR processes, reducing manual code review time.*
- *Encourage the use of internal tools for code hygiene and simplification before merging.*
- *Maintain a balance between automation and human oversight to ensure quality in product design and features.*
- *Continuously adapt and refine workflows as AI capabilities evolve to enhance team efficiency.*

What I'm actually going to show today is what I would consider a pretty practical demo of some of the workflows that I think will help uplevel people who are using cloud code. Um, [clears throat] the way I like to describe it and in some of my experience so far in the industry, we're kind of operating in like these really compressed time horizons, but the way that we work at Enthropic because we just have access and we're playing all the time and all day. It feels like we're like always looking for the next way to work and make ourselves more optimal. And so I'm going to show some of the things that have become really popular internally for how we work and hopefully this will all help you up level. One thing I'll call out is that I am a CLI die hard.

Like I've designed the CLI like cloud is in there. I think it's just like like I used to design CLI products even before AI. So know that like this is not how you need to work. The desktop app is much more accessible and like you can do everything that I show today on the desktop app as well. But um this is going to be on an open source repo. Does everyone know what Excaladraw is? Does anyone know what it is? It's like a Yeah, excellent products. Super great that they're open source. You can always contribute if you actually want to practice. They're a really great one because they have a great backlog of issues and a pretty open community.

First tip, if you don't know it, is that you should always be starting your work in a work tree. Does everyone do this or does anyone know what this is? All right. So, work trees are pretty complicated and generally I don't think you need to know what they are. But all you need to know is that if you have a local source of your repository on your computer and you want to run multiple clouds at once, you might have run into the issue where your clouds start conflicting with each other and you're doing one thing in one and another thing in

another and they're overwriting each other and they're competing. Work trees makes an isolated copy of your repository so that you can do multiple parallel tasks. So if you find yourself having multiple windows open and doing multiple things at once or you see your engineers who have four or five clouds open, know that they either have multiple copies of the repository on their laptop and they might call it like repo one, repo 2, repo 3, or they're using work trees. And I recommend work trees cuz it's a little bit easier to manage. And to start it up, you just do cloud-work tree. And so you can kind of see in the branch there, it's in a work tree and it already checked out a new branch for me. So pro tip number one, if you're multi-clotting, use work trees.

The second pro tip I'll give. I know not everyone has access to this depending on your organization. So I just say like I am always in opus 1 million contexts just so you don't have to think about it. And then I am always in fast mode. Uh [laughter] yeah. So once again, I know it's not accessible to everyone >> and I am going to do this so that we can fast forward through the demo a little bit because I only have 15 minutes, but I do think that it does, you know, it does make a little bit of a difference.

Okay. And then I'm a big typo writer. Claude is actually really good at getting typos if you type them, but uh I put this here so that y'all could see it. This is a prompt that I'm just going to get Excal to add a new autocomplete feature. That's it. No design specs, no just like I want to add autocomplete. I want to see what it looks like. A few things I'm going to call out here that I think are important when you're prompting. The first is I built myself a /prototype skill. I asked Claude to build this. You could build in like 2 seconds. All it does is get Claude to generate uh n number of options default to five of a different implementation of a feature. Generated an HTML file, preview it, and then iterate on it a few times. And I can enter that n and I can enter what the feature is going to be.

Uh, I did not write this. No one ever writes their skills by hand anymore. If anyone tells you they do, they're lying. Everyone just prompts them. Um, and then a few things I will add in here is I always ask Claude to pick an option for me first. I think it's actually fun to see what Claude comes up with. So, historically, I used to be like, "Oh, like prototype and then I'll choose." Now, I'm like, "Oh, prototype and then you should choose what you're going to do and then tell me why." And then I say another one is like, "Feel free to research online." Now, if I were doing this in my production codebase, I'd say like please look at Slack, Google Docs, any discussions in like like Bitquery, like look at all these sources and figure out which is the best one. But given that this is an open source repo, online research is fine. And then, you know, implement the best option, verify, get the styles to match. Um, really important one, put up a PR with a screenshot. Cloud is actually really good at this. And I actually don't review the outputs anymore of Cloud in the Transcript. I'm typically reviewing a PR that cloud put up that has a recording of the feature it implemented.

And then a new feature that we released I think a few months ago is loop. Does anyone has anyone used this before? Loop is pretty much just cycle keep going. So loop until you're done just means keep going at this until it's fully done. And so I'll show some more applications of loop in the future. But it's like a pretty standard prompt. If an engineer asked me to like build something, this is like well to be honest they would probably do this on their own without me. and then send me a vibecoded prototype. But if I was doing it from the ground up, this is how I would do it. Oh, the other pro tip that I forgot to mention, I'm always and everyone at

Anthropic is always in auto mode. So auto mode is our kind of workaround for lower permission modes.

We have a classifier that will detect whether or not a risky action is being taken on your computer. So you don't have to be sitting there being like, "Yes, I accept. Yes, yes, yes, yes." So you should be using it in this way and it'll go a lot faster. Now, I've done a lot of these demos before and cloud takes a long time to work. So, typically if I were doing this, I'd actually spin up a bunch of these. But because we're here and I'm trying to showcase a workflow tonight, I'm actually going to show a few other things I do on the side while Cloud is working. But, um, I think these are a few tenants that I hold true as I'm working today and they're going to kind of inform some of the workflows that I'm going to show all of you today. So the first is that Claude tragically and most LLMs are not good at design yet. So what this means is that you should still very much be in the loop for the craft and the decision-m. It doesn't mean this is going to be holding true forever, but a lot of my workflows revolve around the idea that I still need to be the one making a decision about what actually goes into the product. Perfect. So you can see here, sorry this is going to be a little back and forth, that Claude has used my prototype skill to list a bunch of different options of what tab to complete could look like. And so it's showing all the different ways I'll typically like preview. Sometimes I'll play around with it. Um, sometimes Claude asked me for one. Does anyone have a preference on one that they like?

[snorts] >> Two. >> All right. I'm going to go two. That's it. All right. Um, the second is that I actually, yes, coding is automated, but I hand off so much of my work that is not coding to Claude. And if you're not doing that, then you're actually not using Cloud. in the most I think like automated way. You we really need to start thinking about more than just code when we think about AI automation. And then the third one which I don't think is controversial for this room but is definitely been one that has been an uphill battle a lot of places is just because everyone can ship doesn't mean not everything should ship. And that's a pretty important distinction that we need to start making as we're adding more features and as everyone has the ability to access code and like push a production. We need better systems that scale.

And so some ways that we're fighting this internally today is I actually use Claude in the web. Does everyone have this internally at their teams or companies? Uh you might have it with your own internal infrastructure, but I have like claude in the cloud or cloud in the web as a service that we offer. And I use it as a way to send like hundreds of tiny little polish fixes all the time that I find in the app. They're not worth spinning up a new session for.

They're not worth dedicated time. I just have all of these going. And I would say uh sometimes I get complaints from the engineers that's there's too many of them. So every now and then I ask claw to squash them all into one PR for me and sometimes they just get auto approved because they're tiny CSS changes and no one really needs to review them. So I would say like this is a really great way to maintain craft and quality in your product and it's something that you can do on the side as cloud is working.

A second one is that almost everyone at Enthropic actually always has clouds running and always has clouds running to help merge PRs. So, one thing that I never do anymore actually is once an idea is done, I'm never actually in CI. I'm never reviewing like like until merge. I'm never like addressing code review comments and

stuff like that. It's actually all fully automated now. And so, there's two tricks that I do to do that. The first is All right, perfect. So, now you can see I asked Cloud to verify.

So, Claude is opening Chrome right now and it's going to test that it's working, but I'm just going to We can watch it if you want. If you guys have the Claude in Chrome, I would highly recommend using it cuz if you're doing front-end changes, it's actually the best way to have Claude be able to self verify. And you can see the ADHD we all way we all work right now where we're like doing one thing and then Cloud pops in and is doing another thing. Uh, but I'm just going to let that go. Uh, these are the commands. The first one I do actually before when I make big code changes, it's just a really good way to like do a little bit of a hygiene check.

uh simplify and code review are internal to our repository but I guarantee you if your engineering team is using AI they have an equivalent of that that kind of prunes your codebase so ask your engineering partners like how like what skills do you guys have to simplify your PRs and then they'll be able to put them up and then this commit push PR is one that runs a bunch of checks that we have internally as well so these are all ones that our team has built some of them are available externally but I also like know that all your engineering teams have dev prod teams that have built these and you should be using them uh and then the second One is one that almost everyone engineering non-engineering has at anthropic. This is built into a skill but I wrote out the full command that we used to use which is just to review any open PRs and just get them over the finish line. And the somewhat annoying thing but really great thing that we have is because it's connected to Slack, it will DM people if they are on your code review as well or it'll DM the stamp channel and ping who's on call. So the full integration of your suite is where it's at.

Uh and then the last one which I think has been the most gamechanging for me personally as our product suite has grown is actually I have a cloud code routine which is a schedule task if you use uh cloud co-work and what this does is it scrapes all of our repositories for any front-end changes that anyone has implemented. It will check if a designer was involved in it by going through Slack, Google Meet transcripts, like uh Google Docs, like any kind of data that it has access to. If a designer was not involved in it, it will flag that it shipped without a designer or not. [laughter] If it's shipped without a designer, it will come up with an adversarial or it'll review the design and then it will typically come up with an adversarial design in a PR already drafted, DM it to the engineer who shipped that feature and then ask them to work with the designer on that feature.

I tested this a few times and I had to turn off the DMing because cloud was actually really bad at design unfortunately and we did have to like it's and like that goes back to my first one like cloud is not good at design yet but I think this is the level of automation that you have to be thinking like not just the first step but like the next step the next step the next step after that like you want to be pushing it so far that it's like building the actual end product and I think we have a lot of forgiveness right now in the industry because we're all testing how these workflows work so uh people were very forgiving when I first tried this and it was really bad and now I just consume this myself, but uh I'm always ready for when the next model is going to come out and this is actually going to be ready and available because I have it all written up. So when we're testing the next one, I'll just throw that up again and see if it'll work and it typically gets better.

Yeah. And so these are the three tips if I could like really impress on you of like workflows that we're using that have really like just augmented how the team works. These are the three things I demoed today. And then if we go back to the session, you can see that Claude is now recording a screenshot from Claude in Chrome here. Uh, it does it through a series of GIFs. It's going to post it up and then it's going to open the PR.

Yeah, but that's it for my demo. Hopefully this is helpful and hopefully this helps you guys all use cloud code.