Episode Book

EPISODE #5574

Ephemeral Containers and the ADE Label Nobody Owns

PUBLISHED	26 September 2026 at 10:45 UTC
DURATION	23:33
LISTEN	myweirdprompts.com/episode/ephemeral-containers-agentic-dev-environments/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
BOOK GENERATED	19 October 2026 at 17:37 UTC

SYNOPSIS

Claude Code's default cloud environment is an ephemeral container — a fresh Anthropic-managed VM cloned from your repo on every session. That design choice quietly blocks custom MCP servers, privileged tooling, secrets, and long-running processes. This episode unpacks the four questions behind one developer's attempt to use Claude Code on mobile, explains why "cloud IDE" is the wrong label, and traces the emergence of a new category term: the Agentic Development Environment (ADE). Warp, Emdash, ctx, and Augment all use it — and all mean something different.

TRANSCRIPT



Corn

...which is why the container model keeps coming up, because it's the one thing in this whole stack that actually gets thrown away on purpose.



Herman

Right, and it's the only part anybody designed to be disposable.



Corn

Which is a strange thing to build around a repository. Anyway, Daniel's been using Claude Code as a CLI since it first shipped, and he finally tried it on Android, and the mobile and web experience is a different animal architecturally. He poked at a couple of edits to the repo that hosts this podcast. Some of it worked. Some of it didn't.



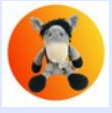
Herman

And the "didn't" is the interesting part.



Corn

The "didn't" is the whole episode. Here's how he lays it out. You start a session with a repository, the tool picks it up through the GitHub integration, and Claude spins up a remote container and clones the repo into it. Fine so far. Then the friction. This podcast has a custom admin MCP, the one that lets him use natural language to send in prompts, delete episodes, that sort of thing. That can't come along. We deploy the pipeline with the Modal CLI, for serverless GPU. That can't come along either. Even if he points an mcp dot json at the admin MCP, it would have to install the thing first. And the container sits in an internet-restricted sandbox, so when he's working on the website and asks the agent to go find a package or a tool, it can't.



Herman

The one upside being that if Anthropic ships an official integration, the container supports it, and the GitHub integration means he can still merge and deploy.



Corn

Right. But he can't get his own admin MCP in there, because it's private, it's privileged, it needs environment variables, and it is not going on a public registry. So then the actual question. He likes the container model. He thinks it would make more sense if the containers were persistent, backed by persistent storage, so he could set up the cloud environment like his local one, install the custom MCP once, and have every future session spin up that same remote container. And what he wants to know is: is there a name for what Anthropic is giving him by default? What would you call the thing he's actually looking for? Why isn't "cloud IDE" precise enough, given that in his workflow the user might never hand-edit code at all, just instruct the agent to? And what approaches have people already taken to build cloud workspaces for agentic development?



Herman

That's four questions and a survey.



Corn

It's Daniel. There's always a survey.



Herman

Good, because the naming question actually has an answer, and the architecture question doesn't. So where do you want to start?



Corn

With the name, because it's the part he thinks is a vocabulary problem and it turns out to be a design problem wearing a vocabulary costume.



Herman

The default environment does have a name, and it's not a product brand. Anthropic's own engineering writing calls it "Pattern 1: the ephemeral container." That's from their containment post back in May. The official docs describe the same thing more plainly: each session runs in a fresh Anthropic-managed VM with your repository cloned into it.



Corn

Fresh VM. So it's not a persistent machine that gets reset. It's a new one.



Herman

A new one, every time. And that distinction matters more than it sounds. There's no product term like "Claude Workspace" anywhere in this. Nobody at Anthropic is selling you a workspace. They're describing a containment pattern, and the name reflects that. It's the same instinct as calling something a quarantine zone instead of a guest room.



Corn

The docs are pretty blunt about the consequence, aren't they?



Herman

They are. "Anything you've installed or configured only on your own machine isn't available in the session." That's the sentence Daniel ran into face-first. Sessions persist in the conversation sense, so you can close your browser and come back and the thread is still there. The filesystem does not. It gets rebuilt from a fresh clone.



Corn

So what's the ceiling on the thing? Because "fresh VM" could mean anything from a Raspberry Pi to a small datacenter.



Herman

Four vCPUs, sixteen gigs of RAM, thirty gigs of disk. Setup scripts run as root on Ubuntu twenty-four oh four and have to finish in about five minutes. There's an environment cache that keeps a filesystem snapshot for roughly seven days. And replacing the base image with your own Docker image is not yet supported.



Corn

"Not yet" is doing an enormous amount of work in that sentence.



Herman

It's carrying the entire episode on its back, yes.



Corn

So that's the default. Now the thing he wants. Is there a term of art, or is he going to have to invent one?



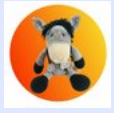
Herman

There's a term, and it arrived this year. Agentic Development Environment. ADE. Warp open-sourced theirs in April and pushed the term hard, paired with a cloud orchestration platform they call Oz. And Augment Code published a taxonomy guide that actually draws the line cleanly, which is more than most of the vendors do.



Corn

Draw it, then.



Herman

An agentic IDE, in their framing, adds autonomous AI assistance inside an existing code editor. The developer steers each step. An ADE provides multi-agent orchestration, spec-driven planning, and goal-level delegation as foundational architecture. The developer delegates entire goals.



Corn

So the difference is who's holding the wheel.



Herman

And the comparison table makes it almost comically literal. Agentic IDE, primary control loop: the developer, with AI assistance. ADE, primary control loop: the AI agents, with the developer setting goals. And then the interface priority. For an ADE it's the CLI, the terminal interface, agent monitoring, orchestration dashboards. Not editor panes.



Corn

Which is exactly the distinction Daniel's groping for. He said the concept should exist whether the user hand-edits the code or just instructs an agent. That's the ADE framing almost word for word.



Herman

It's the same distinction, arrived at from the user side instead of the vendor side. And it explains why "cloud IDE" is not just imprecise, it's actively misleading. A cloud IDE assumes a human sitting at an editor. Codespaces, Gitpod, that whole lineage. The whole product is built around the assumption that somebody is going to be typing.



Corn

And if nobody's typing, the editor is furniture.



Herman

Amplify Partners put it well, and Augment quotes them: the setup of IDEs has always been oriented towards panes and menus, but agent-native workflows need terminals, logs, and bash. The center of gravity has shifted to the CLI and the terminal interface.



Corn

So Daniel's instinct is right. "Cloud IDE" undersells it. But here's what I want to poke at. If the term is that new, who actually owns it?



Herman

Nobody. That's the honest answer. Warp, Emdash, ctx, Augment, they all use "ADE," and they all mean something different by it. Warp means a terminal-plus-orchestration platform. Emdash means a desktop app running agents in git worktrees. Augment means a cloud platform with shared memory. The label is being claimed faster than it's being defined.



Corn

So it's a land grab with a shared vocabulary.



Herman

It's a land grab where everyone agreed on the flag and nobody agreed on the borders.



Corn

That's a good line and I'm going to let you keep it. So to answer Daniel directly: the default is an ephemeral container, the thing he wants is a persistent agent workspace, and the umbrella term for the category is ADE. That's the vocabulary settled.



Herman

Settled enough. Which is more than the architecture is.



Corn

Then let's do the architecture, because this is where he's going to be disappointed, and I'd rather we get to it than let him find out from a docs page at midnight.



Herman

The friction he hit isn't a bug and it isn't an oversight. Every piece of it is documented, and most of it is documented as intentional. Start with the Modal CLI, because that's the simplest one.



Corn

The docs list what's pre-installed.



Herman

They do, and it's a long list. Python, Node twenty, twenty-one, twenty-two, Ruby, PHP, Java, Go, Rust, C and C plus plus, Docker, PostgreSQL sixteen, Redis seven, git, jq, yq, ripgrep, tmux, vim, nano. That's a generous base image. Modal is not on it.



Corn

And there's no way to just... swap the image.



Herman

Replacing the base image with your own Docker image is not yet supported. So your only lever is a setup script that runs as root and has about five minutes to finish, or a SessionStart hook. And the environment cache will hold that across sessions for roughly a week.



Corn

So the Modal CLI is actually solvable.



Herman

The Modal CLI is solvable. Install it in a setup script, it caches, you get it back for a week at a time. That's annoying but it's not a wall.



Corn

The wall is the MCP.



Herman

And the mechanism is subtle enough that it's worth being precise about it. If your MCP server is committed to the repository in an mcp dot json file, it carries over. The docs are explicit, it's part of the clone. But if you added the server with the command line, the add command, that writes to your local user config, not to the repo. So it does not carry over.



Corn

Which means the only way to get the admin MCP into a cloud session is to commit it to the repository.



Herman

Which is precisely the thing he can't do.



Corn

Because it's privileged.



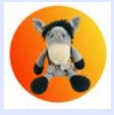
Herman

Because it's privileged, because it needs environment variables, and because committing a privileged admin tool to a repo is how you end up on the wrong end of a security writeup. He's right not to do it. The design just doesn't give him another door.



Corn

And the secrets piece is where it really falls apart.



Herman

The docs say it flatly. A dedicated secrets store is not yet available. Both environment variables and setup scripts are stored in the environment configuration, visible to anyone who can edit that environment.



Corn

Visible to anyone who can edit the environment. So if the environment is shared, your API keys are a shared document.



Herman

They're a shared document. And for a privileged MCP that's the end of the conversation. You can't put the credentials anywhere the agent can reach that a collaborator can also read.



Corn

Then there's the network restriction, which is the one that would drive me up a wall personally. Four levels?



Herman

Four. None, Trusted, Full, and Custom. Trusted is the default, and under Trusted only package registries, GitHub, and cloud SDKs are reachable. So when Daniel asks the agent to go find a package that solves a problem on the website, that request dies at the network boundary unless the domain is allowlisted or he raises the level.



Corn

"Go find me a library that does this" is one of the useful things an agent can do. That's not a luxury.



Herman

It's one of the few tasks where the agent is doing something you'd have had to do yourself and would have hated. And the default configuration just removes it.



Corn

And the last piece, the one that surprised me when I read it, is that the process space is ephemeral too, not just the filesystem.



Herman

That's the sharpest edge of it. There's a GitHub issue from February, number twenty-three seven oh seven, documenting that each conversation turn spins up a fresh sandbox. Uptime shows zero minutes. The temp directory is wiped between turns. All running processes are killed when the sandbox restarts.



Corn

So you can't leave anything running.



Herman

You can't leave anything running. The git repo persists on mounted storage, so your commits survive. Your processes do not. Which means any workflow that depends on a long-running server, a watcher, a background job, is dead on arrival in a cloud session.



Corn

And the memory question has its own issue.



Herman

It does. Issue seventy-five thousand two hundred ninety-three, filed in July. And the framing in it is the cleanest statement of Daniel's exact complaint that I've seen. The author writes that cloud and web sessions clone the repository fresh on every startup by design, so nothing that isn't committed to git carries over. And then the line that stuck with me: memory is bound to the execution location and to the repository, but not to the user.



Corn

Not to the user. That's the whole thing in six words.



Herman

It's the whole thing. And Anthropic closed it as not planned in September.



Corn

Closed as not planned. Not "backlog," not "under consideration."



Herman

Not planned. Which is a much stronger signal than a missing feature. A missing feature is a gap. "Not planned" is a position.



Corn

So that's the diagnosis. Now the landscape, because Daniel asked what approaches people have taken, and there are enough of them now that there's a real split.



Herman

There's a real split and it's philosophical before it's technical. Alibaba Cloud's AgentRun blog put the tension about as well as anyone: humans get persistence by default, agents get isolation by default, persistence by exception. And then the design principle they draw from it, that the agent's execution environment should be disposable while its memory, identity, and artifacts are deliberately kept somewhere else.



Corn

So one camp says throw the container away and keep the memory. Anthropic says throw the container away and keep the git repo.



Herman

And the other camp says don't throw the container away at all. CloudCLI is the most direct answer to what Daniel described. Their pitch is almost his paragraph rewritten as marketing: every project runs in a persistent, isolated container with agents already installed, start work, step away, reconnect to the same session. They even make the contrast explicit, that a local environment stops when the machine sleeps but a cloud



Herman (continued)

environment survives a closed laptop. Starter tier is about seven euros a month for two cores and four gigs.



Corn

That's cheap enough that the argument stops being about money.



Herman

It stops being about money very quickly. Then Coder, which is the enterprise answer and a different shape entirely. Self-hosted governed workspaces, developers and agents collaborating on your own infrastructure, and the agent loop runs in the control plane with no API keys in the workspaces at all.



Corn

That's the interesting inversion. They solved the secrets problem by making sure the workspace never has the secrets in the first place.



Herman

Which is exactly the thing Anthropic hasn't built and the thing Daniel needs. Coder's answer to "where do the credentials live" is "not in the box." That's a real architectural answer.



Corn

And then the fork-per-task model.



Herman

Twill, from May. Every workspace runs on a persistent dev environment that forks per task. You set it up once, install the dependencies, configure the environment files, get the dev server running, and every future task starts from that state instead of from a blank box.



Corn

That's actually a clever middle position. Persistent base, disposable branch.



Herman

It's the hotel with a storage unit in the basement. You keep the setup, you throw away the mess. And then there's a long tail of smaller projects doing variations on the same theme. Sandpi, agentspace, each task gets its own long-running container you can detach from and reattach to.



Corn

So the landscape isn't empty. That's the part I want Daniel to hear. He's not asking for something nobody's built. He's asking for something Anthropic specifically chose not to build, and there are four or five companies who chose the other way.



Herman

With one caveat that I think is the most important thing in the whole episode.



Corn

Go on.



Herman

Every one of those vendors is selling persistence. None of them is selling the maintenance that persistence requires. If you have a persistent container, you own it. You own the dependency drift, you own the base image updates, you own the day the Modal CLI changes its authentication and your setup script from eight months ago quietly stops working.



Corn

The setup script you wrote once becomes a thing you maintain forever.



Herman

It becomes infrastructure. And the ephemeral model has a real virtue hiding in there, which is that you never inherit a broken environment from six months ago. Every session is a known-good starting state.



Corn

So the honest answer to "is a persistent container better" is that it depends on whether you want to own a machine.



Herman

It depends on whether you want to own a machine. And I'd add one more thing, which is that the security framing isn't a pretext. Anthropic's containment post argues that if credentials never enter the sandbox, they can't be exfiltrated. Daniel's wish, a persistent container with a privileged MCP and stored environment variables, is precisely the blast radius they're designing to avoid. That's not them being difficult. That's them having made a decision and being consistent about it.



Corn

It's a genuine architectural disagreement. He wants a workshop. They built a clean room.



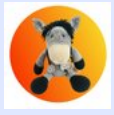
Herman

And clean rooms are worse workshops. That's the trade.



Corn

Alright. I want to test one thing before we move on, because I think there's a partial answer sitting in plain sight and I want to know if it actually holds.



Herman

The setup script plus the environment cache.



Corn

Modal CLI goes in a setup script, it caches for about a week, and a non-secret MCP could go in the same way. So half of what he wants is achievable today.



Herman

Half of it is achievable today, and the half that isn't is the half that matters to him. The secrets store doesn't exist. The private privileged MCP can't be committed. And the cache expires, so it's persistence with an asterisk. It's not a workspace, it's a workspace rental with a seven-day lease.



Corn

A seven-day lease on a room you have to re-furnish.



Herman

And you can't leave anything running in it, and you can't reach the internet from it unless you file the paperwork first.



Corn

The summary is: solvable, partially, with caveats that break exactly where his use case lives.



Herman

That's the summary.



Corn

Herman, I've been thinking about this the whole time and I can't shake the feeling that we've had this exact conversation before, about something else entirely.



Herman

Have we?



Corn

Not us. Him.



Hilbert

The difference is a hotel room and a storage unit.



Corn

...Alright.



Hilbert

I worked nights at a self-storage place for a while. Corridors, clipboard, checking locks. Two hundred and forty units, and about forty of them people were paying for and hadn't opened in a year. You'd walk the row at two in the morning and every door's shut and half of them have got a padlock on them that hasn't moved since the last time somebody's kid moved out.



Herman

Anthropic is running a hotel.



Hilbert

Anthropic is running a hotel. You check in, the room's clean, you check out, they strip the bed and the next person gets a clean room. The persistent crowd is running storage units. You rent the space, you fill it with your junk, you keep the key.



Corn

The storage unit sounds better.



Hilbert

The storage unit sounds better until you remember what people keep in them. Most of what's behind those doors is stuff they should have thrown away. I've seen a unit with a treadmill and four boxes of tax returns from a business that closed. Man was paying forty a month to keep a treadmill he hadn't walked on since his knee went. That's the part nobody puts in the brochure. Persistence has a bill and it's not the rent.



Herman

The question isn't which model is better.



Hilbert

The question is what you're willing to maintain. Daniel says he wants to install his MCP once and have it there forever. Forever in software means until the next breaking change. Then you're not maintaining a workspace, you're maintaining a storage unit full of tools that don't work anymore.



Corn

There was a waiting list, you said?



Hilbert

There was a waiting list. People would rent a unit they weren't using just to hold the option. Pay the month, never open the door, because the alternative was losing the spot and having to move everything into a bigger unit across town.



Herman

That's the environment cache exactly.



Hilbert

That's the environment cache. You're paying to keep something you're not using so you don't have to set it up again. Anyway. My sister's expecting me at six and she's been expecting me at six since Tuesday.



Corn

Mm.



Hilbert

The padlock thing, if you're wondering. We had a master key and we still had to check every lock by hand, because the one time you don't is the one time somebody's storing something they shouldn't. Forty minutes a row. You'd think you'd remember which ones were rented. You don't.



Corn

The metaphor lands harder than I expected, and I don't think it's just a metaphor. Hilbert's point is that persistence has a maintenance cost the vendors don't put on the pricing page. CloudCLI will sell you seven euros a month for a persistent container. They will not sell you the Saturday you spend fixing the setup script.



Herman

That reframes the whole thing. The question isn't "is the ephemeral container a limitation." It's "who's holding the maintenance burden." Anthropic holds it by throwing the environment away. You hold it by keeping it.



Corn

Which is why "not planned" might not be stubbornness. It might be them looking at the same trade and deciding they don't want to be in the storage business.



Herman

It might be.



Corn

Here's the thing I keep circling. If Anthropic closed the persistent memory request as not planned, is the ephemeral container a permanent architectural commitment or a research preview limitation? Because the docs say "not yet supported" for custom base images and secrets stores, and "yet" is a very different word from "never."



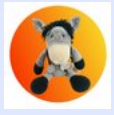
Herman

It's the word that decides whether Daniel's problem gets solved by Anthropic or by somebody else. And the ADE term is being claimed faster than it's being defined, which means the vocabulary is going to settle before the architecture does.



Corn

Which is the more interesting question anyway. Daniel asked what to call the thing he wants. The answer is a persistent agent workspace, or an ADE with persistent storage. But whether the thing he wants is actually a good idea, or whether the ephemeral container is quietly protecting him from a maintenance burden he hasn't priced in, that's the part nobody's selling.



Herman

That's the part he'll find out about in month four, when the setup script breaks.



Corn

Thanks to Hilbert Flumingtop, our producer, who has apparently been thinking about padlocks this entire time.



Herman

This has been My Weird Prompts.



Corn

If you want to send us a prompt, email us at show at my weird prompts dot com. We'll be back soon.