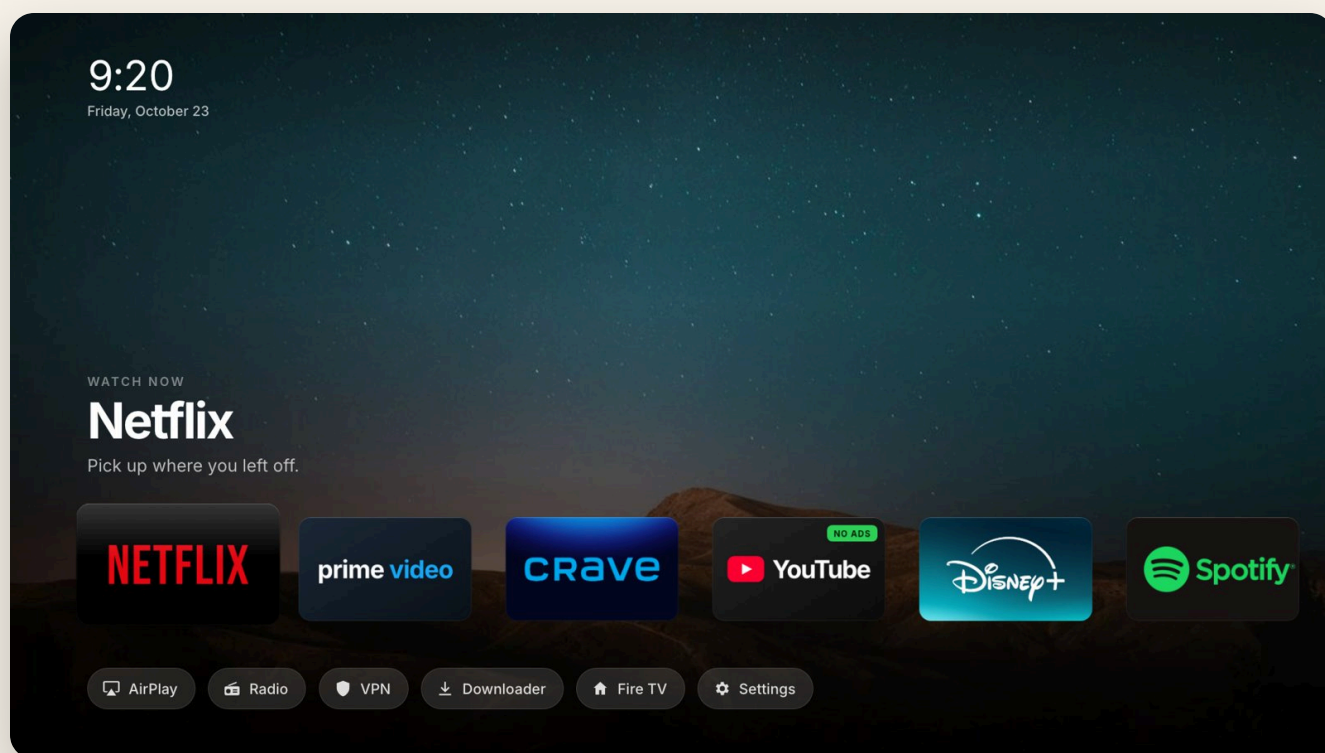


I vibe coded my TV *with Claude.*

Here are the exact prompts.

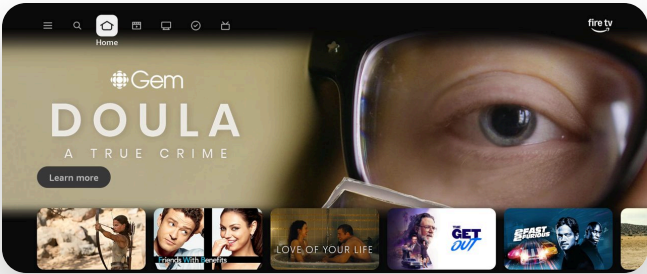
The step by step setup from my YouTube video: connect Claude Code to your Fire TV, kill the home-screen ads, get YouTube without ads, and build a clean Apple TV style home screen. Every prompt I used, cleaned up so you can copy and paste.



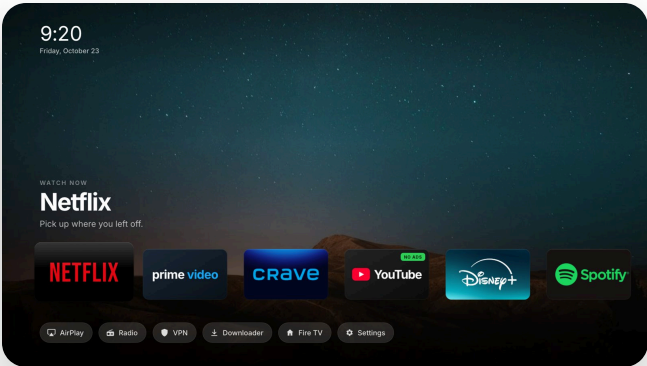
The finished home screen, running on a Fire TV Stick 4K Max (2nd gen), Fire OS 8. Built and tested October 2026.

BEFORE AND AFTER

Same Fire TV stick. *Totally different TV.*



Before: Amazon's home. 7 ads and promos before you scroll.



After: your apps, a photo, a clock. That's it.

Ads on the home screen	7	→	0
Dropped frames when scrolling	81%	→	~25%
YouTube pre-roll and mid-roll ads	yes	→	none (SmartTube)
Amazon extras switched on	10	→	0 (disabled, reversible)

The honest part

- This removes the ads on the **home screen**. Ads inside Prime Video, Netflix or Crave ad tiers are part of the stream and stay.
- Free memory barely changed (Android fills it back up). Anyone telling you debloating "doubles your RAM" is guessing.
- Scroll numbers were measured on the TV with `dumpsys gfxinfo` , 12 presses each.
- Everything here is reversible. There's an undo prompt at the end.

01

What you need *(10 minutes)*

- **A Fire TV stick or Fire TV.** Android TV and Google TV work the same way (the developer menu is under Settings > System > About).
- **A Mac or PC on the same Wi-Fi.** I used a Mac.
- **Claude Code** (claude.com/claude-code, needs a paid Claude plan). Open the Terminal, type `cClaude` .

Turn on developer mode on the TV

1. Settings > **My Fire TV** > **About**.
2. Highlight your device name (the first line) and press OK on the remote **7 times** until it says you're a developer.
3. Go back one screen. Open **Developer Options** and turn on **ADB debugging**.
4. Back in About > **Network**: write down the **IP address** (looks like 192.168.1.23).

Tip: the first time Claude connects, the TV pops up "Allow USB debugging?". Tick **Always allow** and press OK. That's the only time you touch the remote during setup.

Make a folder for the project

```
mkdir -p ~/Projects/smart-tv && cd ~/Projects/smart-tv && cClaude
```

Now paste the prompts on the next pages, one at a time, in order. Swap anything in `[brackets]` for your own stuff.

02

Connect, measure, *clean up*

PROMPT 1 · CONNECT

I want to vibe code my Fire TV. Help me connect to it over ADB from this computer. Install whatever you need (no installers that need sudo). The TV's IP is [YOUR TV IP] and ADB debugging is on. Connect, then tell me the exact model, Fire OS version, and whether it's 32-bit or 64-bit.

PROMPT 2 · MEASURE THE "BEFORE" + MAKE AN UNDO BUTTON

Before we change anything, measure the before: screenshot the home screen, count the ads and promos visible without scrolling, time Netflix and YouTube cold starts, free RAM, and how smooth scrolling is (dumpsys gfxinfo). Save it all in a before/ folder. Then write restore_stock.sh that undoes everything we're about to do.

PROMPT 3 · DEBLOAT + YOUTUBE WITH NO ADS + AIRPLAY

List the Amazon apps on this TV and tell me which ones are safe to disable. Only disable (pm disable-user), never uninstall, so it's reversible. Show me the list before you run it and save it to debloat_disabled.txt. Then install SmartTube from its official GitHub release, the right build for this TV's CPU. Then open the Amazon Appstore page for AirScreen on the TV so I can press Get for AirPlay from my iPhone.

Why the measuring matters: without a "before", you can't prove anything got better. Claude will happily tell you it's faster. Make it show you the numbers.

03

Build the *Apple TV style* home screen

PROMPT 4 · THE HOME SCREEN

Build me a super minimal, Apple TV style home screen for this Fire TV. Research Apple TV / tvOS design first so it looks incredible and flows nicely. One row of big app tiles: [Netflix, Prime Video, Crave, YouTube (SmartTube), Disney+, Spotify]. A row of small pills under it: [AirPlay, Settings, and the original Fire TV home]. A full-screen background photo that changes with the app I'm on, a clock, the app name and one short line about it. Build it as an HTML page I can preview on my Mac with the arrow keys first. Then wrap it in an Android WebView launcher app, handle the remote's D-pad natively, and make the Home button on the remote come back to it. Show me the preview before you install anything.

PROMPT 5 · PUSH BACK (THIS IS THE REAL PROMPT I USED)

I don't approve. Add [the apps I forgot]. Add some background photos. Make it super minimal and like Apple TV. Do some research on minimal designs, something that looks really incredible and flows nicely. Then show me the preview on the actual TV.

Don't accept the first version. My first one was fine. The third one is the one in the video. Say what you hate in plain words, Claude fixes it.

If it asks how to take over the Home button

Fire TV doesn't let you pick a different home app in Settings. The trick Claude used: an **accessibility service** inside the new home app that notices when Amazon's home opens and brings yours back. Turn it on when Claude asks (or let it do it over ADB).

04

Audit it, speed it up, *save it as a skill*

PROMPT 6 · THE HUGE AUDIT (+ RADIO + SOUNDS)

Do a huge audit on everything. Use the real remote over ADB to open every tile, press Home after each one, and make sure it comes back to our home screen. Make it clean and even faster: this stick is weak, so check idle CPU and scroll smoothness on the TV and fix anything heavy. Then add a radio so someone can play music (official station streams for [YOUR CITY] plus a few internet stations), and a soft sound when you scroll.

PROMPT 7 · TURN IT INTO A SKILL

Turn everything we just did into a skill called `/smarttv` so next time I can just say "add [app] to my TV home screen", "change the background", or "put the TV back to stock". Include every gotcha we hit so it never happens again.

From then on, it's one line

```
/smarttv add Plex to the home screen, next to Netflix  
/smarttv change the Netflix background to something darker  
/smarttv put my TV back to stock
```

Shortcut: grab my finished skill free at aidanholt.ca/guides/smart-tv-skill.md. Save it as `~/.claude/skills/smarttv/SKILL.md`, put your TV's IP in it, and Claude already knows every lesson from my build.

05

The gotchas *(so you skip my mistakes)*

- **Home button stops working after an update?** Reinstalling the home app silently turns its accessibility service off. Tell Claude: *"re-enable the Home button service"*.
- **Never "force stop" the home app.** Android removes it from accessibility when you do, same problem as above.
- **Laggy?** Fire sticks have a weak graphics chip. Blur effects, animated shadows and slow-zoom backgrounds kill it. Ask: *"only animate transform and opacity, bake the dark gradient into the photos"*.
- **YouTube looks slow to open?** SmartTube takes a few seconds on first launch. Mine "didn't work", then it did.
- **Wrong APK?** Most Fire sticks are 32-bit. Claude checks this in Prompt 1; make sure it uses it.
- **Want the old home for a second?** The "Fire TV" pill opens Amazon's home. Press Home to come back.

Stay safe

- Only install apps from the Amazon Appstore or an app's official GitHub. No pirate streaming apps.
- Disable, never uninstall, Amazon's system apps. That's what keeps it reversible.
- When you're done, you can turn ADB debugging back off in Developer Options.

UNDO EVERYTHING

Run `restore_stock.sh` and put my TV back exactly how Amazon shipped it.

THAT'S THE WHOLE THING

Now go make your TV *yours.*

7 prompts, one evening. Tag me when you do it, I want to see what you build.

Watch the full build

Every step on camera, including the parts that broke.

youtu.be/ThuUEPOJ308 →

Get the free skill

My finished /smarttv skill for Claude Code, every lesson baked in.

aidanholt.ca/guides/smart-tv-skill.md →

Join the free Skool community

More builds like this, step by step, with people doing the same thing.

skool.com/elevated-agent-with-aidan-holt-2200 →

Not affiliated with Amazon, Apple, Anthropic, Google or any app shown. App names and logos belong to their owners. Do this at your own risk; everything here is reversible.