

Make It Talk Back

Ask a question out loud and hear the answer out loud, so working with Claude Code becomes a conversation instead of reading a screen you are not looking at.

Built by [@brycenwood.ai](#) · October 2026

THE PROBLEM

You already talk to it. It does not talk back.

Dictation solved half of this a long time ago. Most people who use Claude Code heavily stop typing prompts pretty quickly, because talking is faster than typing and you can do it while your hands are busy.

The other half never got solved. The answer still comes back as a wall of text on a screen. That is fine at a desk. It is useless when you are driving, walking a job site, standing in a kitchen, or working in a different window, which for a lot of business owners is most of the day.

The information was ready and I was not in a position to receive it. That is the whole gap. This system closes it, and the entire thing runs on your own machine with no extra subscription and no API key.

THE KEY IDEA

Do not read the message. Read the answer.

This is the part that decides whether you keep the system or turn it off on day two, so it is worth understanding before you build anything.

The obvious version is text to speech. Claude finishes, a program reads the message out loud. Try it once and you will hate it. A real answer is full of headings, bullet points, file paths, and code. Heard out loud, that is unbearable, and the one sentence you needed is buried ninety seconds deep.

The version that works is different. **Claude writes a short spoken summary of its own answer, and only that gets read to you.** Everything else stays on screen for later. You are not listening to a transcript. You are listening to a person tell you what happened, answer first.

WHY THIS ONLY WORKS WITH AN AI ON THE OTHER END

A normal screen reader cannot do this, because it does not know which sentence mattered. Claude does, because Claude wrote the answer. It knows the recommendation, the number you asked for, and the decision you now owe. Asking it to say that part out loud is a small request that changes the whole experience.

STEP 1

Understand hooks, in one paragraph

Claude Code has a feature called hooks. A hook is a command that runs automatically when something specific happens. When Claude finishes answering. When you submit a new prompt. When a session starts. You do not have to be a programmer to use them. You describe what should happen and Claude writes the script and wires it up.

This whole system is two hooks and one small command. That is it.

STEP 2

Build it

Open Claude Code in any folder and paste this in. Answer its questions as it goes. It will explain what it is doing in plain English and tell you exactly what it changed on your machine.

PASTE THIS INTO CLAUDE CODE

I want Claude Code to read its answers to me out loud, so working with you is a conversation instead of me reading a screen. I am not a programmer, so explain what you are doing in plain English as you go.

Build this on my machine:

1. A Stop hook that speaks your final message aloud when you finish a turn. Use whatever text to speech is already built into my operating system. Nothing paid, no API key.
2. IMPORTANT: do not read the whole message. Reading a long technical answer out loud is unbearable. Instead:
 - Let you write a short spoken summary inside a marker in your message, and speak ONLY that, verbatim.
 - If there is no marker, fall back to reading just the opening prose and stop at the first heading or bullet, capped at about 30 seconds.
 - Strip code blocks, file paths, and URLs before speaking, since they are unlistenable.
3. A hook on prompt submission that kills any speech in progress, so when I start talking back it shuts up immediately. It must NOT touch my on/off setting.
4. A command called `voice` I can run from any terminal that controls every session at once:
 - voice off / voice on
 - voice stop (kill the sentence in progress right now)
 - voice status (is it on, which voice, is it speaking)
 - voice test (say a sample line)
 - voice again (replay the last thing it said, for when I miss it)
5. A way for you to speak DURING a turn, not only at the end, so you can tell me something important the moment you find it instead of saving it for the summary. Make it non-blocking.

Handle these three failure cases, because they are the ones that will actually bite:

- If I have two Claude sessions running and both finish at once, they must not talk over each other. Use a lock and queue the second one.
- Make sure the hook reads my FINAL answer, not mid-work narration. Solve this structurally. Do not paper over it with a timing delay, because a delay will look fine and then break on a long turn.
- Make sure `voice stop` genuinely kills speech mid sentence. Prove it by testing it while a long sentence is playing, not just by showing me that it prints a confirmation.

Then write me a test suite that verifies all of it, and make sure the

tests DO NOT speak out loud when they run.

Before you start, tell me your plan and what you are about to change on my machine. Then build it, test it, and tell me exactly how to turn it off if I hate it.

STEP 3

Teach it what is worth saying out loud

Once it can talk, the failure mode flips. It stops being silent and starts being chatty, and a system that narrates everything is one you will mute inside a week. Then you miss the message that actually mattered.

Put a rule in your project instructions so it applies to every session automatically.

ADD THIS TO YOUR CLAUDE.MD

Speak up mid-task for: a question you need answered before you can continue, a finding that changes what I would do or contradicts what I believed, a blocker, bad news (especially money leaking or something broken), and a significant thing shipped and verified.

Do NOT speak progress narration, routine tool results, or anything that is merely interesting.

Say each thing once. If you said it mid-task, reference it briefly at the end instead of repeating it whole.

The closing summary is an audio deliverable. Answer first, short sentences, most important thing up front in case I stop listening. It must be COMPLETE on its own. "The rest is on screen" is useless to someone who cannot look at the screen.

THE RULE THAT MATTERS MOST

Over-talking is worse than silence. If it reads you every routine step, you will stop listening, and then you will miss the one message that was actually important. Fewer, better interruptions beat constant narration every time.

The three bugs you are going to hit

I hit all three. None of them announce themselves, and two of them look like they are working when they are not. Ask for these explicitly and you skip an evening of confusion.

Two sessions talking over each other

If you run Claude in more than one terminal, eventually two finish at the same moment and you get two voices at once, both unintelligible. It needs a lock, so the second one waits its turn instead of interrupting.

It reads the wrong thing

If the hook fires before the final message is finished, it reads mid-work narration instead of your answer. The tempting fix is to add a short delay. Do not. A delay looks like it works and then breaks on the first long turn, which is exactly when you were relying on it. Fix it structurally.

An off switch that quietly does nothing

Mine printed a cheerful confirmation and did not stop anything for a week. A backup cleanup step was doing the real work and hiding the bug. Test your stop command by running it in the middle of a long sentence and confirming the sentence actually dies. A confirmation message is not proof.

BUILD THE OFF SWITCH FIRST

Before the summary logic, before the mid-task speaking, get off, on, and stop working and confirm them. The failure you care about is not that it fails to talk. It is that it talks when you badly need it not to, in a meeting or next to a sleeping kid.

THE PAYOFF

What changes

The obvious win is hands-free. The real win took me longer to notice: **it changes when work is possible.**

Time that used to be dead is now working time. A drive across town, a walk through a shop, ten minutes waiting on someone. You ask a real question about your

business and you get a real answer, out loud, and you make a decision before you have parked.

The test is not whether it is impressive on day one. It is whether you still use it a week later. If you sit at a desk all day looking at your terminal, you probably do not need this. If a lot of your day happens away from a screen, this is the system on this list you will use the most.

That is the whole system.

Zero ad spend. Zero coding experience. The other playbooks live at brycenwood.com/playbook, one download each.

Built by Brycen Wood · [@brycenwood.ai](https://twitter.com/brycenwood.ai) · brycenwood.com