Make your Claude Code limit last with the Advisor

Here's how to set up what I showed in my video. Claude Code has a feature called Advisor. You keep Sonnet as your main model, and when Claude wants a second opinion on a hard decision, it asks Opus. You won't write any code.

What you need

- A paid Claude plan. Claude Code comes with Pro, Max and Ultra (and with Team and Enterprise). It isn't in the free plan. API billing works too.
- The Claude desktop app, or Claude Code in a terminal. The steps run in **Claude Code**, which is the **Code** tab of the app, not the regular chat.
- The Anthropic API, Claude Platform on AWS or Claude Models on Google Cloud (Anthropic

managed). The Advisor isn't available on Amazon Bedrock, Google Cloud's Agent Platform or Microsoft Foundry.

Anthropic labels the Advisor **experimental**: "Behavior, pricing, and availability may change."

What it does

You pick a main model, which does the work, and an advisor model, which Claude consults at key moments. The advisor reads the whole conversation, including every tool call and result, and gives guidance that Claude applies before it carries on.

Claude decides when to call it. Anthropic says it tends to call it before committing to an approach, when an error keeps recurring, and before declaring a task done, but the timing is model-driven and not rule-based.

With Sonnet as the main model and Opus as the advisor, Anthropic describes it this way: Sonnet handles routine work and escalates planning, ambiguous failures and completion checks to Opus.

Step 1: open Claude Code and set Sonnet as your main model

1. Download the Claude desktop app from claude.ai/download, install it and sign in.
2. Click **Code** at the top of the sidebar.
3. Make a new folder to work in, for example "Advisor test" in your Documents.
4. In the row under the message box, choose **Local**, then click the folder control, choose **Open folder...** and pick that folder.
5. Type:

```
/model sonnet
```

The `sonnet` alias means the latest Sonnet model. You can also type `/model` on its own to open a picker.

Step 2: turn on the Advisor

Type `/advisor` on its own to open a picker of the available advisor models, then pick **Opus**. Or skip the

picker and type:

```
/advisor opus
```

Claude Code confirms with `Advisor set to` followed by the model's name. Your choice is saved, so it stays on in your next sessions too. You'll also see a notice that the Advisor Tool (experimental) is on and may use more tokens.

In the desktop app, and in other places with no terminal picker, `/advisor` with no model prints the current advisor and the names it accepts, and `/advisor opus` sets it. This needs Claude Code version 2.1.260 or later.

The advisor has to rank at or above your main model. Anthropic's table accepts Opus as the advisor for a Sonnet main model, so this pairing works.

Step 3: give it a real job

Ask Claude for a job as you normally would. Anthropic says the Advisor fits long, multi-step tasks where most turns are routine and the quality of the plan decides the result, for example a large refactor,

a debugging session where an error keeps coming back, or a task you want checked before Claude says it's done. It adds less on short tasks where there's little to plan.

You can also ask for it, the way you'd ask for any tool:

```
Consult the advisor before you continue.
```

There's no setting to force or limit how often Claude calls it. If you want it more or less often, say so in your instructions.

How to tell it's working

When Claude calls the advisor, the transcript shows an `Advising` line with the advisor's model name. When it finishes, the line says one of three things:

- **Reviewed.** The advisor looked at the conversation. If it gave guidance you can read, press `Ctrl+O` to see it.
- **Declined.** The line reads `Advisor declined to advise on this request`.

- **Unavailable.** The call failed, and the line reads `Advisor unavailable` followed by an error code.

Claude generally follows the advice. If a recommended step fails, or your files contradict it, Claude tells you instead of following it blindly.

What it does to your limit

Each time Claude calls the advisor, Opus reads the conversation. That uses tokens at Opus's rate, on top of what Sonnet uses.

- **On a subscription plan,** advisor usage counts toward your plan's usage limits.
- **With API billing,** you pay the advisor model's rates for those tokens.

Anthropic says Claude calls the advisor at decision points, not on every turn, so pairing a faster main model with a stronger advisor typically costs less than running the stronger model all the way through. Check what you've used with `/usage`.

Anthropic's own test, from its post "The advisor strategy": Sonnet with an Opus advisor scored 2.7

percentage points higher than Sonnet alone on SWE-bench Multilingual, and cost 11.9% less per task. The advisor writes a short plan, typically 400 to 700 text tokens, and Sonnet does the rest of the work at its lower rate. That's one test on one set of tasks, not a promise for yours, and Anthropic gives no figure for how much longer a plan's limit lasts.

Opus is the more expensive model. Anthropic's cost page says Sonnet handles most coding tasks well and costs less than Opus, so keep Opus for complex architectural decisions or multi-step reasoning.

Turn it off

```
/advisor off
```

You can also pick **No advisor** in the `/advisor` picker.

Other pairings

The same docs list others, if you want to try them:

Main model	Advisor	What Anthropic says
Sonnet	Opus	Sonnet does routine work and escalates planning, ambiguous failures and completion checks to Opus
Haiku	Opus	Lowest-cost main model with strong planning. Costs more than Haiku alone, but less than switching the main model to Sonnet or Opus
Opus	Opus	A second Opus reviews the first. For high-stakes tasks where an independent check matters more than cost

Written on 17 November 2026 from Claude Code's own documentation pages on the advisor, model configuration and costs, and Anthropic's post "The advisor strategy". I haven't run these steps myself while writing this guide. Claude changes fast: if a menu has moved, look for the same words in the app.