The Opus 5.5 Prompt Fix Guide

The four prompt habits Claude Opus 5.5 quietly broke, and the exact lines to fix them, word for word.

Why your old Opus prompts are working against you now

Anthropic released Claude Opus 5.5 on September 22, 2026, and priced it about 40 percent cheaper to run than Opus 5 overall: $4 per million input tokens instead of $5, $20 per million output tokens instead of $25, and cache reads down 60 percent, from $0.50 to $0.20 per million tokens. Anthropic also says it generates output more than 30 percent faster than Opus 5.

None of that shows up automatically if your prompts still assume the old model. Opus 5.5 changed its default effort level, added a hard block on showing its own reasoning, and behaves differently when you switch effort mid-conversation. If you're pasting the same prompts you used on Opus 5, you're either leaving savings on the table or hitting a flat refusal you didn't expect.

The promise of this guide is narrow on purpose: four prompt habits, the exact wording for each, and where to put them so they stick.

01

How the new defaults actually work

1. Opus 5.5 defaults to medium effort instead of high. Anthropic's own testing found medium effort on Opus 5.5 matches or beats Opus 5 at high effort on coding and knowledge work, in fewer tokens, so most tasks need no effort instruction at all.
2. When you do change effort mid-conversation, Opus 5.5 keeps the cache instead of re-reading the thread. On earlier models, switching effort partway through a long agent session usually invalidated the cache, which meant Claude re-read everything it already knew before continuing.
3. Thinking mode itself can no longer be turned off, and a 'preserved thinking' safeguard blocks any attempt to extract Claude's internal reasoning through the API. Anthropic built this in for EU AI Act compliance. It applies whether you're prompting through the API, Claude Code, or the chat interface.

02

The one thing almost everyone gets wrong

The mistake isn't a typo or a broken prompt, it's a leftover instruction nobody remembers to delete.

WATCH OUT

If any saved prompt, template, or claude.md rule includes a line like 'show your reasoning' or 'explain your thinking step by step before answering,' Opus 5.5 doesn't quietly ignore it. It returns an explicit refusal. In an automated pipeline expecting a normal response, that refusal can look like the whole run broke, when the actual fix is a one-line delete.

03

The prompts, word for word

REMOVE THIS FROM ANY EXISTING PROMPT OR CLAUDE.MD RULE

```
show your reasoning
explain your thinking step by step before answering
```

ADD THIS TO ANY LONG, MULTI-STEP TASK

```
Before you consider this task done, confirm every item below:
1. [Step one]
2. [Step two]
3. [Step three]
Do not stop until every item is checked off.
```

ADD THIS WHEN SPEED MATTERS MORE THAN POLISH

```
time matters
```

SITES SHOWN IN THE VIDEO

- Claude Opus 5.5 (Anthropic)
  https://www.anthropic.com/claude-opus-5-5

Set it up in Claude Code or Codex

1. For Claude Code: open the CLAUDE.md file in your project root (create one if it doesn't exist yet). Paste the checklist template under a heading like 'Long task rule' so Claude Code reads it automatically at the start of every session in that project.
2. Delete any line in that same file that asks Claude to show or explain its reasoning before answering. Opus 5.5 refuses those requests outright, so leaving the line in just adds a failure point.
3. For Codex: the equivalent file is AGENTS.md in the project root, or a shared one referenced from it. The same two edits apply: add the checklist rule, remove any reasoning-display instruction.
4. For a one-off task in either tool, or in Claude.ai directly, you don't need the file at all. Paste the checklist template or the phrase 'time matters' straight into your message.
5. Test it: give Claude a task with at least three real steps and no other instructions. If it stops after step one and calls the task done, the checklist isn't being read, check that the file is in the project root Claude Code or Codex is actually running against.

💡 **QUICK TIPS**

- Keep the checklist short. Three to five concrete items works better than a long list Claude has to parse on every turn.
- You don't need to request an effort level at all for most tasks. Medium is already the default, and it's tuned to match or beat what high effort used to give you on Opus 5.

04

The commands I actually run

KICK OFF A LONG BUILD WITH A CHECKLIST ATTACHED

```
Build [feature]. Before you consider this done, confirm every item below:
1. Code compiles and existing tests pass
2. New behavior has at least one test
3. Docs or comments updated if the public interface changed
Do not stop until every item is checked off.
```

PUSH FOR A FAST FIRST DRAFT

```
Draft [thing]. time matters, so give me a fast first pass I can revise, not the most polished version.
```

BUMP EFFORT FOR ONE HARD STEP MID-THREAD

```
For this next step only, use high effort. Keep everything else about this conversation as is.
```

ASK FOR AN EXPLANATION AFTER THE FACT, NOT LIVE REASONING

```
Now that you've answered, explain in plain terms why you chose that approach over the alternatives.
```

When it breaks

ERROR	FIX
Claude refuses and mentions it can't share its reasoning process	Remove any 'show your reasoning' or 'explain step by step before answering' instruction from the prompt or claude.md/AGENTS.md file.
A long task stops early, main request looks done but smaller steps are skipped	Add the numbered checklist template to the prompt so Claude has a concrete finish line instead of a judgment call.
Effort change mid-thread seems to slow things down or reread context	Confirm you're actually on Opus 5.5, not an older model. Cache preservation across effort changes is specific to this model.
'time matters' doesn't seem to change anything	Use it on a task where thoroughness was genuinely trading off against speed. On a short, simple task there's little slack for the phrase to reclaim.
Automated pipeline breaks after upgrading to Opus 5.5	Check every saved prompt template for reasoning-display instructions first. That refusal is the most common cause of a pipeline that worked on Opus 5 and fails on Opus 5.5.