ELLM

How to Run Commands from the Chat

Learn to run shell commands directly from chat to streamline testing and building processes. Commands can be auto-run, backgrounded, and interrupted, with outputs and fixes proposed by an assistant.

AUTHOR
Paul Flanders

PUBLISHED
10 Sep 2026

DOCUMENT
InfoHub — Knowledge, news, and updates from EssingtonITS

How to Run Commands from the Chat

Paul Flanders · 10 Sep 2026

Estimated time: 4 minutes
Difficulty: Beginner

Why you'd use this

Fixing a failing test or getting a build green means running things, reading the output and changing code in a loop. The assistant proposes shell commands, you approve them, and the output goes back to it so it can act on the result. Long-running commands move to the background so the conversation is not blocked, and every command is cut off by a timeout so nothing runs forever.

Before you start

Permissions required:

- Ability to run the project's tooling from a terminal on this machine (the same shell and PATH VS Code uses).

You'll need:

- A project with a test or build command, for example `npm test` or `pytest`.

Steps

1. Ask for something that needs execution, for example "Run the tests and fix any failures."
2. A card appears with the proposed command and a **Run** button. Read the command, then click **Run**, or **Reject** to refuse it. A rejected command is reported to the assistant

as not executed.

3. A live output box shows the command's output as it streams, with a running timer.
4. When the command finishes, its exit code and output are sent back to the assistant, which proposes a fix if something failed and then offers to run the command again.
5. To stop approving each command on a trusted project, open the **/** menu and tick **Auto-run commands**. Commands then run as soon as they are proposed, except ones classed as dangerous, which still ask.
6. To interrupt at any point, click **Stop** next to the composer. Every command the assistant started, including background ones, is killed.

What you should see

A command that runs longer than 20 seconds shows a "Backgrounded" line; the assistant continues with the output so far and receives the final result when it lands. If the assistant runs more than 25 commands in one task, it pauses with a **Continue** button so a loop cannot run unattended. Commands are generated for your operating system and shell, including remote hosts when you use Remote-SSH, WSL or a dev container.

Troubleshooting

- **"command not found" for a tool that works in your terminal:** the tool is on a PATH your shell sets up in its profile. The extension augments PATH from common locations, but a tool installed somewhere unusual needs a symlink or an absolute path.
- **A command timed out:** raise `ellmCode.commandTimeout` (seconds) in settings. Timeouts are treated as unresolved, never as passes.
- **A result arrives after you pressed Stop:** it is shown as "Result held" and included with your next message rather than restarting the run.
- **The assistant claims tests pass without running them:** it is nudged to run them. If you see a "Nudge" line about an unverified claim, that is this check working.

Related guides

- How to Stop, Pause and Continue a Run
- How to Understand the Verification Gates
- How to Tune Generation Settings for Your Model