Mozout

Generated on 28/11/2026 21:53

KNOWLEDGE BASE · PUBLIC DOCUMENT

Why a coding model must accept tools (function calling)

Choosing a Tool & Fixing Errors

CODE	VERSION	COUNTRY · LANGUAGE
RLX-INT-KB-9864	v2026.11.28	International · English

MozOut International
RLX Holdings (Pty) Ltd. · Group RLX Holdings
Tax no. (SARS) 943 907 6234 · Registration no. (CIPC) 2025/667953/07

https://www.my.mozout.com/knowledgebase/9864

Why a coding model must accept tools (function calling)

Choosing a Tool & Fixing Errors Created on **November 28, 2026** 2 min read



An AI coding tool does more than write text: it reads files, edits them and runs commands on your computer. The tool does the work, but the model decides what to do, and asks for it through tools (function calling). A model that cannot ask for tools can only chat.

How it works

1. The tool sends the model the conversation and the list of tools it has. OpenCode, for example, has `read`, `edit`, `write`, `grep` and `shell`.
2. Instead of answering with text only, the model asks for a tool: for example, to read a file.
3. The tool runs the request on your computer and sends the result back to the model, in a new request.
4. This repeats until the work is done. Each round is one API request, with its own tokens.

Our API accepts tools in all three formats: `tools` in the Chat Completions format (Cursor, Cline, OpenCode, Kilo Code), `tool_use` and `tool_result` blocks in the Anthropic format (Claude Code), and `function_call` and `function_call_output` in the Responses format (Codex).

What happens without tools

The agent answers, but does not edit files or run commands. It looks as if all is well, because there is an answer, but the work never shows up in the project. The fix is to switch to a model that supports tools.

The rounds are also why tools make many requests in a row: a task that reads and edits several files makes one request per round. If you see `429 rate_limited`, see **why agents reach the limit**.

Which models support tools

The six **recommended models** do: `moonshotai/kimi-k2.7-code`, `deepseek-ai/deepseek-v4-pro`, `zai-org/glm-5.2`, `qwen/qwen3.8-max`, `deepseek-ai/deepseek-v4-flash` and `zai-org/glm-5.3-flash`. The model block in **the OpenCode article** has the 79 that support tools, from the most expensive to the cheapest.

The `GET /v1/models` list shows every text model, including those without tools, and does not say which ones support them. In Kilo Code, which reads that list, pick one of the recommended models.

A `sandbox_` key cannot confirm this.
In the Sandbox the model is not called and the answer is always "Resposta de teste (sandbox).". To see whether a model edits files, give it a small task with the `prod_` key, such as creating a test file, and check what it spent in the History tab of **All in One AI**.

What about Aider?

Aider works differently. According to **its documentation**, it asks the model for the changes written in the answer, as whole files or search and replace blocks. The recommended models work in Aider too.

RECOMMENDED

Want to code with All in One AI models? Visit the Coding with AI page.

See Coding with AI →

SEE ALSO

- The recommended coding models →
- Errors when setting up a tool →
- OpenCode with All in One AI →
- Which coding tool to use →
- Coding with AI in All in One AI →

Mozout

MozOut International

RLX Holdings (Pty) Ltd. · Group RLX Holdings

TAX NO. (SARS)	943 907 6234
REGISTRATION NO. (CIPC)	2025/667953/07
REGISTERED OFFICE	1111 Olive Street Saint Louis 63101, Missouri, United States
CONTACT	info@mozout.com · +27 78 823 3325 · www.mozout.com
OPENING HOURS	Monday to Friday - 08 AM to 04 PM Saturday & Sunday - Closed

Online version of this document

RLX-INT-KB-9864 · v2026.11.28 · Generated on 28/11/2026 21:53