Mozout

Gerado a 28/11/2026 22:52

BASE DE CONHECIMENTO · DOCUMENTO PÚBLICO

Porque é que um modelo para programar tem de aceitar ferramentas (function calling)

Escolher a ferramenta e resolver erros

CÓDIGO	VERSÃO	PAÍS · IDIOMA
RLX-MZ-KB-3259	v2026.11.28	Moçambique · Português

MozOut Moçambique
RLX Holdings, S.A. · Grupo RLX Holdings
NUIT 401 191 607 · NUEL 105 045 974 · CAE 62022

https://www.meu.mozout.co.mz/knowledgebase/3259

Porque é que um modelo para programar tem de aceitar ferramentas (function calling)

Escolher a ferramenta e resolver erros Criado em **28 de novembro de 2026** 2 min de leitura



Uma ferramenta de programação com IA não se limita a escrever texto: lê ficheiros, edita-os e corre comandos no seu computador. Quem executa é a ferramenta, mas é o modelo que decide o que fazer, e pede-o através de ferramentas (function calling). Um modelo que não sabe pedir ferramentas só consegue conversar.

Como funciona

1. A ferramenta manda ao modelo a conversa e a lista das ferramentas que tem. O OpenCode, por exemplo, tem `read`, `edit`, `write`, `grep` e `shell`.
2. Em vez de responder só com texto, o modelo pede uma ferramenta: por exemplo, ler um ficheiro.
3. A ferramenta executa o pedido no seu computador e manda o resultado ao modelo, num pedido novo.
4. Repete-se até o trabalho acabar. Cada volta é um pedido à API, com os seus tokens.

A nossa API aceita ferramentas nos três formatos: `tools` no formato Chat Completions (Cursor, Cline, OpenCode, Kilo Code), blocos `tool_use` e `tool_result` no formato da Anthropic (Claude Code) e `function_call` e `function_call_output` no formato Responses (Codex).

O que acontece sem ferramentas

O agente responde, mas não edita ficheiros nem corre comandos. Parece que está tudo bem, porque há resposta, mas o trabalho não aparece no projecto. A solução é trocar para um modelo que aceite ferramentas.

É também por causa das voltas que as ferramentas fazem muitos pedidos seguidos: uma tarefa que lê e edita vários ficheiros faz um pedido por volta. Se aparecer `429 rate_limited`, veja **porque é que os agentes chegam ao limite**.

Que modelos aceitam ferramentas

Os seis **modelos recomendados** aceitam: `moonshotai/kimi-k2.7-code`, `deepseek-ai/deepseek-v4-pro`, `zai-org/glm-5.2`, `qwen/qwen3.8-max`, `deepseek-ai/deepseek-v4-flash` e `zai-org/glm-5.3-flash`. O bloco de modelos do **artigo do OpenCode** traz os 79 que aceitam ferramentas, do mais caro ao mais barato.

A lista de `GET /v1/models` mostra todos os modelos de texto, também os que não aceitam ferramentas, e não diz quais aceitam. No Kilo Code, que lê essa lista, escolha um dos recomendados.

Uma chave `sandbox_` não confirma isto.
Na Sandbox o modelo não é chamado e a resposta é sempre «Resposta de teste (sandbox).». Para ver se um modelo edita ficheiros, dê-lhe uma tarefa pequena com a chave `prod_`, como criar um ficheiro de teste, e veja o gasto na aba «Histórico» do **All in One AI**.

E o Aider?

O Aider trabalha de outra forma. Segundo a **documentação dele**, pede ao modelo as alterações escritas na resposta, como ficheiros completos ou blocos de procurar e substituir. Os modelos recomendados servem também no Aider.

RECOMENDADO

Quer programar com os modelos do All in One AI? Veja a página Programar com IA.

Ver Programar com IA →

VEJA TAMBÉM

- Os modelos recomendados para programar →
- Erros ao associar uma ferramenta →
- OpenCode com o All in One AI →
- Que ferramenta de programação usar →
- Programar com IA no All in One AI →

Mozout

MozOut Moçambique

RLX Holdings, S.A. · Grupo RLX Holdings

NUIT	401 191 607
NUEL	105 045 974
CAE	62022
SEDE	Bairro 7 de Setembro Vilankulo 1304 · CEP 0411-02, Inhambane, Moçambique
CONTACTOS	info@mozout.co.mz · +258 84 635 1111 · +258 86 635 1111 · www.mozout.co.mz
HORÁRIO	2ª. à 6ª. Feira - 08 à 16 Horas Sábado & Domingo - Fechado

Versão online deste documento

RLX-MZ-KB-3259 · v2026.11.28 · Gerado a 28/11/2026 22:52