目　录

magpie 项目介绍与使用指南

magpie 项目介绍与使用指南

magpie 项目介绍与使用指南

> *项目地址：yetone/magpie*
>
> *官网与下载：usemagpie.ai　|　文档：usemagpie.ai/docs/start*
>
> *核对日期：2026-11-23。本文依据仓库 main 分支 README、GitHub 仓库元数据（语言、许可证、star 数）与官方文档入口整理，未做本地安装实测，具体行为以实际版本为准。*

一、项目简介

magpie 是一个跑在本地的 AI agent 模型网关 + 菜单栏管理应用，口号是「Every agent's model. One place.」：Claude Code 跑在 Kimi 上、Codex 跑在 DeepSeek 上、Gemini CLI 跑在 GLM 上、OpenCode 用你的 ChatGPT 订阅，全部从一个菜单栏图标切换。

它做的事可以概括为一句话：在 agent 和模型之间插一个本地网关，把「用哪个模型」这件事从每个 agent 各自的配置文件里抽出来统一管理。

项目	情况
仓库	yetone/magpie
主语言	Go
许可证	MIT
创建时间	2026-09-23
热度	约 40.3k stars、3.7k forks（2026-11-23）
官网	usemagpie.ai
运行平台	macOS、Windows、Linux（菜单栏 App / 窗口 / TUI / Web UI / CLI），并支持 Docker 与 Termux

与 Codex 的关系

本文归档在 Codex 目录下，但 magpie 并不是 Codex 的插件或分支，而是一个独立的上层工具，Codex 只是它支持的几十个 agent 之一。

- 它通过改写 Codex 自己的配置文件来生效，不接管 Codex 的会话与工具调用。
- Codex 换成第三方模型后，仍然是 Codex 本体在工作，magpie 只负责转发与翻译请求。
- 反过来，也可以让别的 agent 用上 ChatGPT 订阅里的模型。
- 不想被接管时，卸载脚本会把 agent 配置还原成安装前的样子。

二、要解决的问题

- 配置碎片化：每个 agent 都有自己的配置文件、键名和 base URL 格式，换一次模型要改一遍，还得小心别把注释和格式弄乱。
- 订阅被锁在单个 agent 里：付了钱的 Claude、ChatGPT、Copilot 订阅，通常只能在自己的 CLI 里用。
- 配额断流：某个账号额度用完，agent 直接报错中断，需要人工切换。

magpie 的应对方式：本地起一个网关（默认 `127.0.0.1:3425` ），所有 agent 指向它，由它决定这一次请求落到哪个 provider、哪个账号、哪个模型。

三、整体结构

```
Claude Code / Codex / Gemini CLI / Cursor CLI / Cline / ...
        |  各自原生协议
        ▼
   magpie gateway  127.0.0.1:3425
    ├── 协议互译：OpenAI Chat / OpenAI Responses / Anthropic / Gemini
    ├── 路由：多账号、routing group、intent routing
    ├── 隐私脱敏（可选）
    └── 计量：token、成本、配额、会话
        |
        ▼
Provider：API Key / 本地模型（Ollama、LM Studio）/ 订阅（Claude、ChatGPT、Copilot…）
```

四、主要能力

1. 任意模型配任意 agent

点开一个值，弹出可搜索的模型列表，按 `provider/model` 命名（例如 `deepseek/deepseek-v4-pro` ）。选中后 magpie 会原子地改写该 agent 的配置文件，并保留原有的注释、键顺序与格式。

Profiles（配置档） 把每个 agent 的当前设置整体存成一个名字（如「预算」「专注」），需要时一键全套切换；每个 agent 自己的候选模型短名单也会跟着存下来。

2. Provider 与订阅

- 内置大量预设：Anthropic、OpenAI、Google Gemini / Vertex AI、Amazon Bedrock、DeepSeek、Kimi、智谱 GLM、MiniMax、阶跃 StepFun、LongCat、Qwen、百度千帆、腾讯云、华为云 MaaS、火山方舟、Mistral、Voyage AI、Groq、xAI、OpenRouter、Vercel AI Gateway、Together、Fireworks、SiliconFlow、NVIDIA NIM、ModelScope、Ollama、LM Studio 等，任何 OpenAI / Anthropic 兼容 URL 也能手动添加。
- 模型清单从 models.dev 拉取，新发布的模型下次刷新即可选到，名称与 reasoning 档位自动补全。
- 支持从 CC Switch、Claude Code、Codex、Alma 导入已有 provider。
- 每个 key、每个账号可以单独走自己的代理。

订阅即 provider：已登录的 Claude、ChatGPT / Codex、GitHub Copilot、Gemini Code Assist、Antigravity（走本机 agy CLI）、Grok（SuperGrok）等，都能作为 provider 提供给其他 agent。其中 Claude 是驱动本地真实的 `claude` 二进制，并通过 MCP 桥接工具调用。

3. 配额耗尽自动接续

- 同一订阅可以挂多个账号，按需切换。
- Daily warm-up：可以设定时间，提前把 Claude、ChatGPT 账号的 5 小时窗口打开，或在最后一个窗口重置时自动开始。
- Routing group（路由组）：把一组模型打包成一个「虚拟模型」，如 `group/daily-coding`。

模式	行为
`smart`	在还有额度的订阅里，优先用最快重置的那个
`order`	顺序使用，前一个答不了再下一个
`rotate`	每轮换一个成员
`usage`	优先用使用量最少的
`pace`	优先用「每小时周剩余额度最多」的账号

补充规则：

- 一段对话会粘在回答它的那个账号上，只要供应商的 prompt cache 还值钱就不轻易换。
- 组里可以再套组，路由决策在 Routing 页面可见。
- Intent routing：用一个小模型先读这一轮的请求，把测试类任务丢给强模型、把快问快答丢给便宜模型。
- 注意：只有一个账号、且没有配置路由组时，没有任何东西可切，配额错误会照常抛给 agent。

4. 隐私脱敏（默认关闭）

agent 常把 `.env`、配置文件、日志里的 token 一起塞进 prompt。打开 Settings → Privacy 后：

能力	说明
两种识别方式	规则（API key、私钥、token、邮箱、校验位正确的卡号与证件号）；可选本地小模型（约 1.6 GB，跑在 CPU 上）识别姓名、地址、密码
全链路还原	真值换成 `{{API_KEY_k3v9x2mq}}` 这类占位符，回复里再换回来，流式文本与 tool 调用参数都会还原，所以 agent 执行命令时用的是真 key
不破坏缓存	同一个值永远得到同一个占位符，对话、prompt cache 和 thinking 签名不会失效
不落盘	真值只放内存；记录里只保留类型、长度和前几位字符
失败即拦截	开启后若脱敏无法运行，请求会被拦下（503），不会原样发出去
可审计	Insights → Masking 逐条列出被改写的请求，并排显示供应商实际收到了什么
自定义	密钥与个人数据可分别开关，支持更细的类型（护照、IBAN、MAC、公网 IP…）与自定义词和正则，每个值可选 Mask / Warn / Block

代价是：模型只看到 `{{API_KEY_…}}`，所以「这个 token 为什么被拒」这类需要看真值本身的任务做不了，这类值应放进 Never mask。

5. 统一网关与协议互译

网关同时讲 OpenAI Chat Completions、OpenAI Responses、Anthropic Messages、Gemini 四种接口，并在它们之间互译，流式输出、tool call、reasoning 内容都在内。任何「只支持填 base URL」的工具都能直接接：

```
export OPENAI_BASE_URL=http://127.0.0.1:3425/v1
export OPENAI_API_KEY=magpie

export ANTHROPIC_BASE_URL=http://127.0.0.1:3425
export ANTHROPIC_API_KEY=magpie

export GOOGLE_GEMINI_BASE_URL=http://127.0.0.1:3425
export GEMINI_API_KEY=magpie
```

6. 用量、成本与配额

视图	内容
Tokens and cost	token、缓存读写、reasoning、调用次数，按 list price 折算为美元或人民币， 可给任意模型自定义价格
Quota	每个 key、套餐、账号的余额与配额窗口（`magpie quota`）， 窗口将重置而额度大量剩余时会提醒
Sessions	每个 agent 会话及其成本与标题，一键在终端里重开
Context	每次请求的上下文占用、缓存命中比例、由什么占满
Bills	按账号、按上游 key 汇总，支持 OTLP 导出到自建可观测栈

7. 插件与网关中间件

```
magpie plugin add opencode-gemini-auth    # npm 上的 OpenCode 插件
magpie plugin login google-plugin         # 插件自己的登录流程
```

插件跑在 Bun 上（首次使用时自动下载）。除认证插件外，还能写网关中间件——一段能看到所有 agent 请求与响应的 JavaScript：`onRequest` 可以改写或拒绝请求，`onEvent` 看到每个流事件，`onResponse` 看到完整回复；钩子抛错或超时则请求保持原样。

```
// alias.middleware.js
export function onRequest(body, ctx) {
  if (body.model === "fast") {
    return { ...body, model: "deepseek/deepseek-chat" };
  }
}
```

社区已有一批对标 New API 的中间件，JSON 配置格式也保持一致：

包	作用
`param-override`	对标 `param_override`，按条件增删改请求字段，或直接拒绝请求

包	作用
`model-map`	对标 `model_mapping`，换名转发，回复仍用客户端请求的名字
`system-prompt`	给所有请求或指定 agent / 模型注入系统提示词
`word-guard`	敏感词过滤，对用户输入与回复都可以拦截或打码
`think-tags`	从回复中移除思考标签，或把 `reasoning_content` 塞进正文

8. 共享、多机与同步

- 局域网共享：打开 Share on local network，为每个客户端发一个命名网关 key，可设每日 / 每周 / 每月的 token 或费用上限。
- Remote magpie：笔记本直接使用桌面上那台 magpie 的 provider、账号与分组，同时仍然接线本机的 agent。
- Docker：`ghcr.io/yetone/magpie` 跑在服务器或 NAS 上，用 Web UI 管理。
- 同步与备份：备份成文件，或通过 WebDAV（坚果云、Nextcloud…）、S3、加密的 GitHub 仓库在多机之间同步。

9. 其它细节

- Library：instructions、MCP server、skills、subagents 写一次，magpie 按各 agent 自己的格式写进对应文件，删除时只删自己写过的那部分。
- 给不能联网的模型加搜索：可接 Tavily、Brave、Exa、Firecrawl、You.com、DeepSeek、火山方舟豆包搜索、腾讯云元宝搜索、TinyFish、SearXNG 等，按拖动顺序询问；同一个 API 可挂多个 key 轮流用，最近被拒的 key 排到最后。也能作为 MCP server 挂给任意 agent，提供 `web_search` 与 `web_fetch` 两个工具。
- 图像与视频：给任意 agent 提供 MCP 图像工具（模型自选）；视频需要 Grok 订阅或 MiniMax API key。
- agent 版本管理：Agents 页显示各 CLI 的版本，按安装方式升级，并给出未安装者的安装命令。
- WSL：Windows 下 WSL 各发行版里的 agent 也会被接管，会话也会被统计。
- 保持屏幕常亮、深浅色主题、Omarchy 主题适配。

五、安装

```
# 1. 官网下载安装包：https://usemagpie.ai
# 2. 一行脚本（macOS / Linux）
curl -fsSL https://usemagpie.ai/install.sh | sh

# 3. 包管理器 / 源码安装
nix profile install github:yetone/magpie-releases
go install github.com/yetone/magpie@latest
```

几点说明：

- macOS 构建已签名并公证；每个发布文件都带 GitHub Actions 构建证明（`gh attestation verify -R yetone/magpie-releases`），每个版本自动更新。
- Linux 上脚本装桌面版；缺少 WebKitGTK 4.1 时会先用系统包管理器安装（会要 sudo 密码）；没有桌面或没有终端可询问时，改装终端版并提示缺哪个包。可用 `--gui` / `--tui` 指定。
- 有防火墙时用 `--proxy` 或 `--mirror`。

- Docker 镜像为 `ghcr.io/yetone/magpie`，Termux 也可安装。
- 卸载：`curl -fsSL https://usemagpie.ai/install.sh | sh -s -- --uninstall`，会先把各 agent 的配置恢复原样，再移除脚本装的东西；`~/.config/magpie` 里的设置默认保留，加 `--purge` 才清除。

六、快速上手

图形界面三步

1. 加 provider：Providers → Add provider，选预设粘 key，或用订阅登录。
2. 选模型：在 Agents 页给每个 agent 选一个模型。
3. 开一个新会话，已经在用新模型了。

命令行等价操作

```
magpie provider add deepseek sk-...            # 预设只需 key
magpie claude deepseek/deepseek-v4-pro         # Claude Code 用 DeepSeek
magpie codex moonshot/kimi-k2.5                # Codex 用 Kimi
magpie group add "Opus anywhere" \
  models=claude/claude-opus-5-5,copilot/claude-opus-5.5 routing=smart
magpie claude group/opus-anywhere              # 几个订阅之间自动接续
magpie save work && magpie use work            # 保存 / 切换配置档
magpie quota                                   # 各套餐还剩多少
magpie settings set theme dark                 # 设置项也能从命令行改
magpie tool generate_image "a magpie, watercolour" -o hero.png
magpie tui                                     # 在终端里跑整个界面
```

七、数据与配置位置

- 设置目录：`~/.config/magpie`（卸载时默认保留）。
- 网关监听：`127.0.0.1:3425`。
- magpie 自身不额外中转内容，prompt 与回复只发往你配置的 provider。

八、隐私与遥测

magpie 每天会匿名上报一次「正在被使用」：随机 id、版本与系统、用了哪些 agent / provider / 模型（自己添加的 provider 只上报为 `custom`）、装了哪些 Discover 插件、用了 Library 市场里哪些 MCP server（按市场 id），以及合作方在添加面板与插件页的曝光、打开、添加次数。不上报名称、URL、账号、key、prompt 与用量。

- 可在 Settings → Privacy 部分或全部关闭，或设 `DO_NOT_TRACK=1`。
- 合作方链接走 `usemagpie.ai/go/…` 计点击，图标走 `usemagpie.ai/api/icon` 代理，服务端不保留 id 与地址。
- 为发现订阅，magpie 会读 Claude Code、Codex 的登录文件与 macOS Keychain。可在 Settings → Privacy → Subscription accounts 关闭，或执行 `magpie accounts discovery off`；单次关闭用 `MAGPIE_NO_ACCOUNTS=1`。API key 方式的 provider 不受影响。

九、注意事项与限制

- 不接受 Pull Request：仓库只有维护者能提 PR。有 bug 或功能需求要去 Discord 描述，由维护者实现。想贡献代码的人需要提前知道这一点。
- Issue 数量很大：截至 2026-11-23 约 4 960 个 open issues，说明项目活跃，也说明反馈未必能及时响应。
- 功能面很宽：网关、脱敏、计量、插件、MCP、Library、共享、同步都在一个应用里，属于「瑞士军刀」型工具，轻量需求可能觉得重。
- 默认不脱敏，需在设置里手动开启；脱敏也可能误判，应到 Masking 页核对。
- 单账号 + 无路由组 = 没有 failover，这是设计如此，不是 bug。
- 开源不等于完全自建：核心与安装脚本是 MIT，但官网、导入链接、图标代理、Remote 等依赖 usemagpie.ai 的服务。
- 名字撞车：与 Apache Magpie、Apache 软件基金会无关。

十、常见问题

我只用一个 agent，还需要它吗？

不需要。README 自己说：如果你只用一个 agent、只用官方套餐就够，那不必装。它面向的是「一个以上 agent、一个以上 provider、或一个以上账号」的场景。

装了之后 agent 的配置会不会被改坏？

magpie 采用原子写入，并保留注释、键顺序与格式；卸载时会把 agent 配置还原。仍建议在重要环境安装前先备份配置。

为什么某个模型在列表里没有？

模型清单来自 models.dev，新模型可能要等下次刷新；也可以手动添加 provider 或自定义模型价格。

能不能多台机器共用一套 provider？

可以：局域网共享（发带限额的命名 key）、Remote magpie（用另一台的 provider 与分组），或 Docker 部署在 NAS 上通过 Web UI 管理。

它和 New API / one-api 是一回事吗？

定位有重叠但不完全相同：magpie 是桌面端 + 本地网关，重点在「管理本机多个 agent 的配置」和「把订阅变成 provider」；它对标 New API 的部分主要体现在网关中间件（ `param-override` 、 `model-map` 等）。需要多人多租户的后台管理时，New API 这类方案仍在它之上。

十一、总结

magpie 的价值是把「agent 用什么模型」从散落的配置文件里抽出来，集中成一个本地网关：

- 对个人：一份订阅多种用法，配额见底自动切换，模型可以随便试。
- 对团队：统一出口、统一计量、统一脱敏，按人发限额 key。