


OpenClaw安装与使用教程



1. OpenClaw 是什么

 OpenClaw（原 Clawdbot/Moltbot）是一款开源 AI 智能体，核心是**用自然语言指挥电脑执行真实操作**，实现从“理解意图→规划任务→落地执行→反馈结果”的完整闭环。它的应用场景覆盖**个人效率、办公协作、开发运维、垂直行业**四大类：

命名寓意：“Open”代表开源社区驱动；“Claw”象征灵活强大的执行能力。

核心优势：高可塑性、全模型适配、跨平台协同、主动执行、持久记忆，核心架构由 Gateway、Agent、Skills、Memory 四部分组成。

OpenClaw的应用场景：

一、个人效率与生活自动化（最易上手）

• 办公自动化

- 邮件管理：自动分类、提取信息、生成回复、定时发送。
- 文档处理：生成报告 / 简历 / 会议纪要，批量格式转换、提取要点。
- 日程管理：创建日程、设置提醒，自动同步会议邀请。
- 信息整理：从网页 / PDF / 图片提取文本，整理为表格 / 清单。
- 周报 / 日报：自动汇总工作内容，生成结构化周报。

• 日常事务处理

- 文件管理：批量重命名、分类整理、自动备份。
- 信息查询：自动查天气、航班、快递，汇总行业资讯。
- 定时任务：自动关机、备份、下载订阅内容。

- 报销自动化：OCR 识别收据，自动分类并更新报销表。
- 个人知识管理：整理碎片信息，构建个人知识图谱。

二、企业办公与团队协作（全行业通用）

• 团队协作

- 任务管理：同步任务清单、跟踪进度、发送催办提醒。
- 协作文档：自动汇总成员文档，统一格式并共享。
- 会议支持：生成议程、记录纪要、自动分发、同步日程。
- 工作简报：定时汇总进展，生成团队简报并推送。

• 行政与 HR

- 简历筛选：自动解析简历，匹配岗位要求。
- 新员工入职：提供入职指引、资料收集、流程自动化。
- 差旅 / 报销：自动处理差旅申请、费用核算、报销流程。

三、软件开发与 IT 运维（开发者专属）

• 开发辅助

- 代码审查：自动审查 PR，检查规范、漏洞、优化点。
- 代码重构：辅助遗留系统跨语言重构、代码优化。
- API 文档：自动同步 API 文档，保持代码与文档一致。
- 数据库迁移：辅助 Schema 迁移、数据校验、回滚预案。
- 代码搜索：全局语义搜索，快速定位代码片段。

• 运维自动化

- 基础设施：生成 Terraform 脚本、K8s 配置、Nginx 规则。
- 日志分析：监控 K8s 异常日志，自动告警与定位问题。
- CI/CD 优化：自动调优流水线，提升构建与部署效率。
- 云成本优化：巡检云资源，识别闲置与浪费，给出优化建议。
- 异常检测：实时监控业务指标，异常自动告警与初步排查。

四、垂直行业定制化应用（深度落地）

• 内容创作 / 新媒体

- 热点监控：自动扫描全网热点，生成社交媒体内容。
- 内容分发：长视频自动切片、适配平台、添加标签、排期发布。

- 品牌监控：24 小时监控品牌提及，情感分析，及时处理差评。
- 多平台运营：自动管理微信、微博、抖音等矩阵账号。
- **电商零售**
 - 竞品监控：自动抓取竞品价格、库存、活动信息。
 - 订单处理：自动同步订单、发货、售后、评价管理。
 - 数据报表：自动生成销售、库存、用户分析报表。
- **金融 / 投资**
 - 市场监控：7×24 小时监控行情、新闻、公告，自动预警。
 - 数据整理：自动汇总财报、研报，生成投资分析报告。
 - 交易辅助：执行预设交易策略，自动下单、止盈止损。
- **法律 / 财税**
 - 合同处理：自动生成、审查、比对合同，提取关键条款。
 - 票据处理：OCR 识别发票、单据，自动分类、记账、报税。
 - 法规检索：自动检索相关法规、判例，生成法律意见书。
- **教育 / 培训**
 - 助教助手：自动批改作业、答疑、生成学习报告。
 - 资源管理：自动整理课件、题库、试卷，智能分发。
 - 学习规划：根据学生情况，自动生成个性化学习计划。
- **工业 / 物联网**
 - 设备巡检：驱动巡检机器人 7×24 小时作业，自动上报异常。
 - 数据采集：自动采集工业传感器数据，实时分析与预警。
 - 流程控制：优化生产流程，降低能耗，提升效率。

2. OpenClaw 安装

支持 macOS、Linux、Windows、树莓派，新手优先选 npm 安装或一键脚本，开发者可选源码安装，安装前完成简单前置准备即可。

2.1 前置准备（必做）

一、安装 Node.js

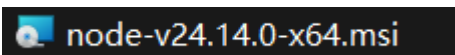
1. 下载地址: <https://nodejs.org/en/download>

The screenshot shows the Node.js download page for Windows. At the top, it says "Download Node.js®". Below that, there are dropdown menus for "v24.14.0 (LTS)", "Windows", "using Docker", and "with npm". A green banner below the dropdowns says "Info: Want new features sooner? Get the latest Node.js version instead and try the latest improvements!". Below the banner is a code block with instructions for installing Node.js using Docker. The code is as follows:

```
1 # Docker has specific installation instructions for each operating system.
2 # Please refer to the official documentation at https://docker.com/get-started/
3
4 # Pull the Node.js Docker image:
5 docker pull node:24-alpine
6
7 # Create a Node.js container and start a Shell session:
8 docker run -it --rm --entrypoint sh node:24-alpine
9
10 # Verify the Node.js version:
11 node -v # Should print "v24.14.0".
12
13 # Verify npm version:
14 npm -v # Should print "11.9.0".
```

Below the code block is a "PowerShell" button and a "Copy to clipboard" button. At the bottom, there is a section for "Or get a prebuilt Node.js® for Windows running a x64 architecture." with two buttons: "Windows Installer (.msi)" and "Standalone Binary (.zip)". A red arrow points to the "Windows Installer (.msi)" button with the text "点击下载安装".

2. 下载的软件安装包:



3. 双击安装, 一直下一步就可以

4. 验证安装结果

```
C:\Users\admin>npm -v
10.9.2

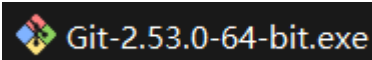
C:\Users\admin>node -v
v22.14.0
```

二、安装Git

1. 下载地址: <https://git-scm.com/install/windows>

The screenshot shows the Git for Windows installation page. The page title is "git --distributed-even-if-your-workflow-isnt". The main heading is "Install". Below the heading, there are tabs for "Windows", "macOS", "Linux", and "Build from Source". The "Windows" tab is selected. The page content includes a link to download the latest (2.53.0) x64 version of Git for Windows. Below that, there is a section for "Other Git for Windows downloads" with links for "Standalone Installer", "Git for Windows/x64 Setup.", "Git for Windows/ARM64 Setup.", "Portable ('thumbdrive edition')", "Git for Windows/x64 Portable.", and "Git for Windows/ARM64 Portable.". A red arrow points to the "Git for Windows/x64 Setup." link.

2. 下载的软件安装包



3. 双击安装，一直下一步就可以

4. 验证安装

```
C:\Users\admin>git --version
git version 2.35.1.windows.2
C:\Users\admin>
```

三、windows权限设置

代码块

```
1 Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

```
PS C:\WINDOWS\system32> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
执行策略更改
执行策略可帮助你防止执行不信任的脚本。更改执行策略可能会产生安全风险，如 https://go.microsoft.com/fwlink/?LinkID=135170
中的 about_Execution_Policies 帮助主题所述。是否要更改执行策略？
[Y] 是(Y) [A] 全是(A) [N] 否(N) [L] 全否(L) [S] 暂停(S) [?] 帮助 (默认值为“N”)： y
PS C:\WINDOWS\system32>
```

Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
这句代码的作用



Qwen3.5-Plus

这条 PowerShell 命令的作用是：**允许当前用户运行本地创建的脚本，以及由受信任发布者签名的远程脚本。**

2.2 安装方式

openclaw官网：<https://docs.openclaw.ai/zh-CN>

中文社区版：<https://clawd.org.cn/>

代码块

```
1 # 全局安装 (使用国内镜像)
2 npm install -g openclaw-cn@latest --registry https://registry.npmmirror.com
3
```

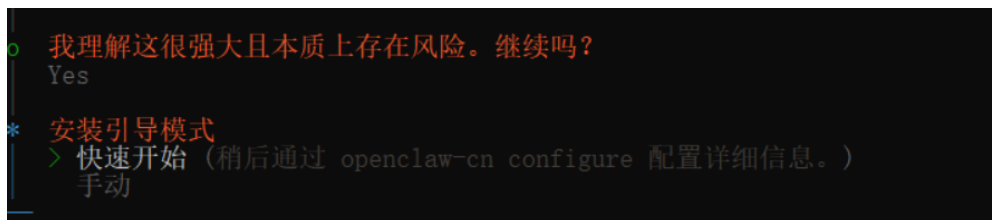
```
4 #检查安装的脚本
5 openclaw-cn --version
6
7 # 运行向导
8 openclaw-cn onboard --install-daemon
```

点击键盘的左右箭头，选择yes后敲回车



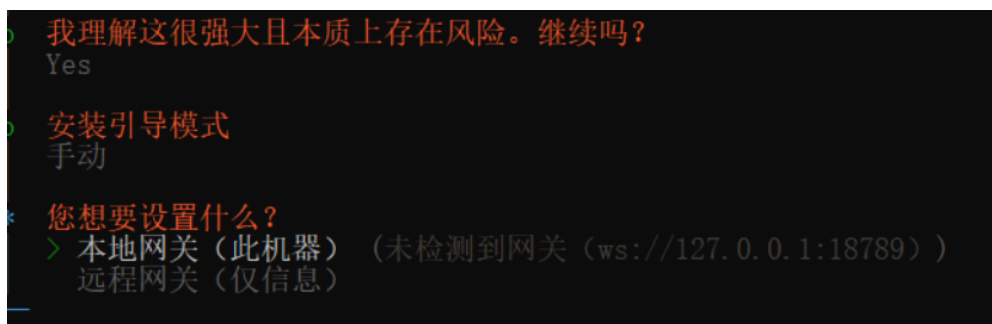
```
终端: https://docs.clawd.bot/windows
OpenClaw 安装引导
安全
安全警告 — 请阅读。
OpenClaw 是一个业余项目，仍处于测试阶段。可能会有粗糙之处。
如果启用了工具，此机器人可以读取文件并执行操作。
不良提示可能会诱使其执行不安全的操作。
如果您不熟悉基本的安全和访问控制，请不要运行 OpenClaw。
在启用工具或将系统暴露给互联网之前，请寻求有经验的人士帮助。
推荐的基础设置：
- 配对/白名单 + @提及门控。
- 沙箱 + 最低权限工具。
- 将机密信息保留在智能体可访问的文件系统之外。
- 对于使用工具或不受信任收件箱的机器人，请使用最强大的可用模型。
定期运行：
openclaw-cn security audit --deep
openclaw-cn security audit --fix
必读: https://clawd.org.cn/gateway/security.html
* 我理解这很强大且本质上存在风险。继续吗?
  Yes / > No
```

敲上下箭头，选择“手动”后敲回车



```
* 我理解这很强大且本质上存在风险。继续吗?
  Yes
* 安装引导模式
  > 快速开始 (稍后通过 openclaw-cn configure 配置详细信息。)
  手动
```

选“本地网关”



```
* 我理解这很强大且本质上存在风险。继续吗?
  Yes
* 安装引导模式
  > 快速开始 (稍后通过 openclaw-cn configure 配置详细信息。)
  手动
* 您想要设置什么?
  > 本地网关 (此机器) (未检测到网关 (ws://127.0.0.1:18789))
  远程网关 (仅信息)
```

默认工作区目录，直接敲回车

```
o 我理解这很强大且本质上存在风险。继续吗?  
Yes  
o 安装引导模式  
手动  
o 您想要设置什么?  
本地网关（此机器）  
* 工作区目录  
C:\Users\82634\.openclaw\workspace
```

选择deepseek模型，敲回车

```
* 模型/认证提供商  
OpenAI  
Anthropic  
MiniMax  
Moonshot AI (Kimi K2.5)  
Google  
OpenRouter  
Qwen  
Z. AI (GLM-5)  
Copilot  
Vercel AI Gateway  
OpenCode Zen  
Xiaomi  
Synthetic  
Venice AI  
Cloudflare AI Gateway  
硅基流动 (SiliconFlow)  
阿里云百炼 (DashScope)  
> DeepSeek (OpenAI兼容 · API key)  
火山引擎 (VolcanoEngine)  
Skip for now
```

敲回车后如图

```
o 我理解这很强大且本质上存在风险。继续吗?  
Yes  
o 安装引导模式  
手动  
o 您想要设置什么?  
本地网关（此机器）  
o 工作区目录  
C:\Users\82634\.openclaw\workspace  
o 模型/认证提供商  
DeepSeek  
* DeepSeek 认证方法  
> DeepSeek API key  
返回
```

选择“DeepSeek API key”后回车



此时，要求输入deepseek官网的API key，所以我们需要登录deepseek找到自己的APIkey，打开[deepseek官网](https://deepseek.com)



创建自己的APIkey



注意：

1. 创建后一定记得复制出来保存一下，因为只能复制一次，以后就不能复制了。
2. 记得充值，不充值的话，后边是无法调用大模型的

将我们复制好的APIkey，粘贴到cmd窗口里

```
o 我理解这很强大且本质上存在风险。继续吗？
   Yes
o 安装引导模式
   手动
o 您想要设置什么？
   本地网关（此机器）
o 工作区目录
   C:\Users\82634\.openclaw\workspace
o 模型/认证提供商
   DeepSeek
o DeepSeek 认证方法
   DeepSeek API key
* 输入 DeepSeek API key
   sk-08a09727 115
```

敲回车后如图，选择“保持当前（deepseek/deepseek-chat）”

```
o 输入 DeepSeek API key
   sk-08a0972797664b60a01d2ac2e6545115
o Model configured -----+
   Default model set to deepseek/deepseek-chat |
-----+
* 默认模型
  > 保持当前（deepseek/deepseek-chat）
   手动输入模型
   deepseek/deepseek-chat
```

端口不需要改，直接回车

```
o Model configured -----+
   Default model set to deepseek/deepseek-chat |
-----+
o 默认模型
   保持当前（deepseek/deepseek-chat）
* Gateway port
   18789
```

选择“Loopback (127.0.0.1)”，回车

```
Gateway port
18789

Gateway bind
> Loopback (127.0.0.1)
  LAN (0.0.0.0)
  Tailnet (Tailscale IP)
  Auto (Loopback → LAN)
  Custom IP
```

选择“Token”，回车

```
Gateway bind
Loopback (127.0.0.1)

Gateway auth
Off (loopback only)
> Token (Recommended default (local + remote))
  Password
```

选择“Off”，回车

```
Tailscale exposure
> Off (No Tailscale exposure)
  Serve
  Funnel
```

这里需写一个网关名称，可以随意写，我写的是 MyGatewayToken，如图

```
Tailscale exposure
Off

Gateway token (blank to generate)
MyGatewayToken
```

回车后，提示开始配置聊天通道，选择“yes”后回车

```
o 通道状态 -----+
Telegram: 未配置
WhatsApp: 未配置
Discord: 未配置
Google Chat: 未配置
Feishu: 未配置
Slack: 未配置
Signal: 未配置
iMessage: 未配置
Feishu: 安装插件以启用
-----+
* 现在配置聊天通道吗?
> Yes / No
```

这里我们选择飞书

```
* 选择一个通道
Telegram (Bot API)
WhatsApp (QR link)
Discord (Bot API)
Google Chat (Chat API)
> Feishu (Lark Open Platform) (插件 • 安装)
Slack (Socket Mode)
Signal (signal-cli)
iMessage (img)
已完成
```

选择使用本地插件路径

```
* 选择一个通道
Feishu (Lark Open Platform)

* 安装 Feishu 插件?
从 npm 下载 (@openclaw-cn/feishu)
> 使用本地插件路径 (C:\Users\82634\AppData\Roaming\npm\node_modules\op
暂时跳过
```

选择“飞书国内版”



此时，提示要输入飞书AppID



我们需要打开飞书开放平台，登录自己的账号

<https://open.feishu.cn/app>



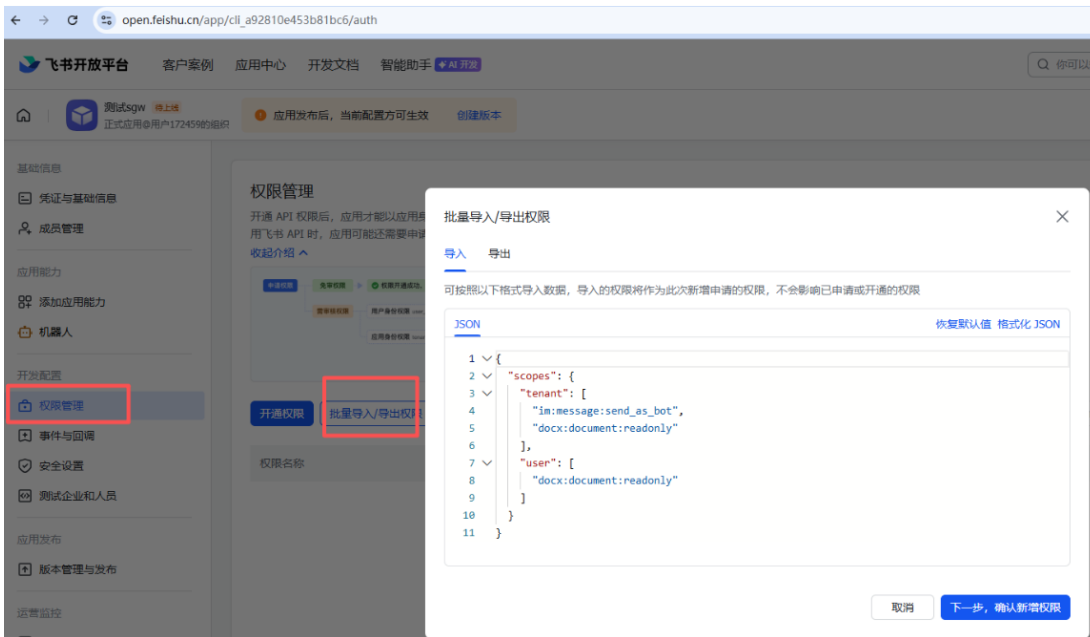
这里信息随便填



添加机器人

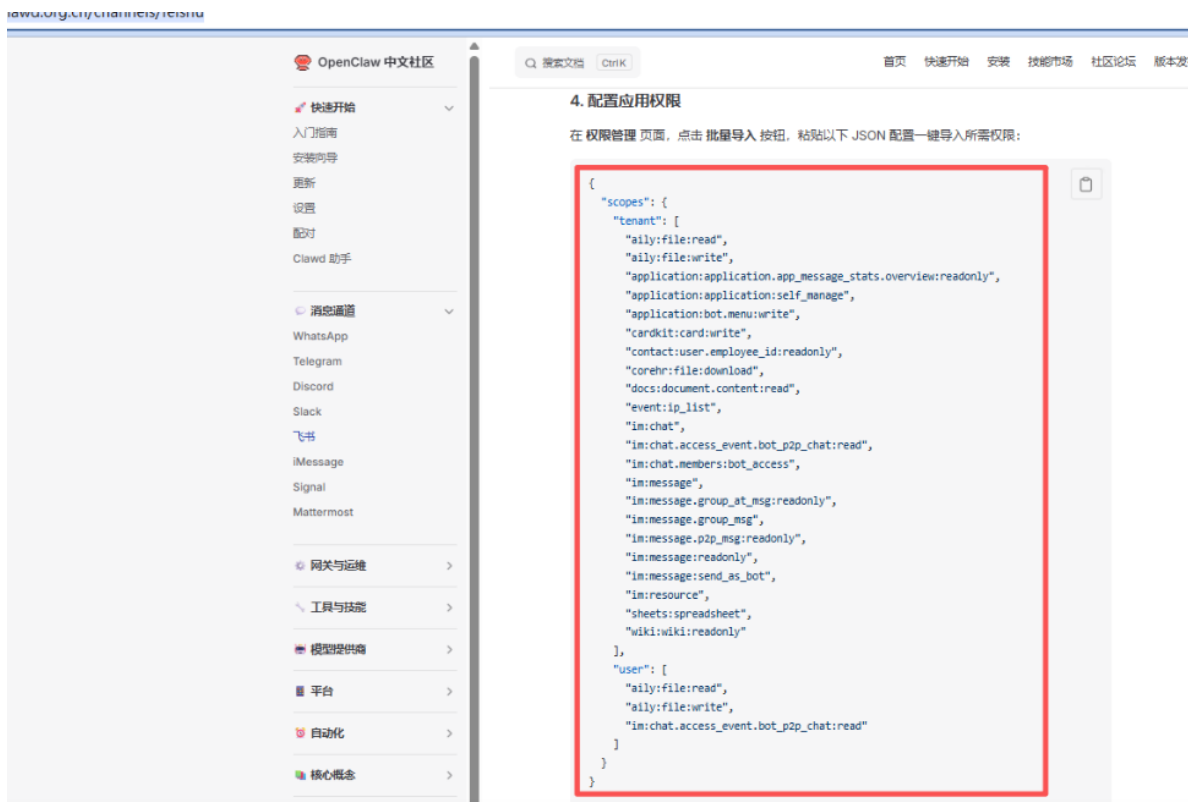


权限管理→批量导入权限

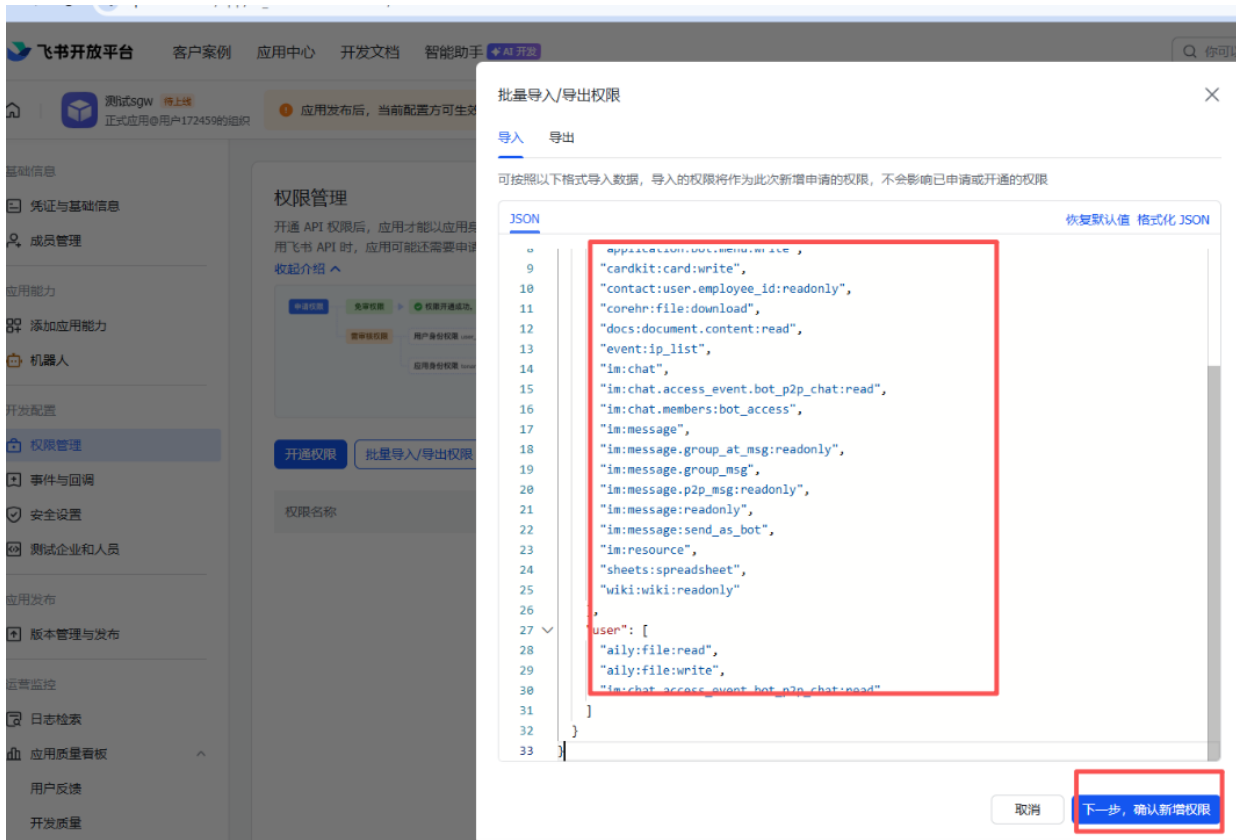


打开飞书官网，寻找权限的json格式数据，权限官网，打开后，往下划动页面，复制如下内容

<https://clawd.org.cn/channels/feishu>



粘贴进来（注意，要先删除之前的配置，然后再粘贴进来）



申请开通



点击“创建版本”

基础信息

凭证与基础信息

成员管理

应用能力

添加应用能力

机器人

开发配置

权限管理

事件与回调

安全设置

测试企业和人员

应用发布

版本管理与发布

运营监控

权限管理

开通 API 权限后，应用才能以应用身份 (tenant_access_token) 或用户身份 (user_access_token) 调用飞书 API；以应用身份调用飞书 API 时，应用可能还需要申请对应的数据权限。[了解更多](#)

[收起介绍](#)



开通权限 批量导入/导出权限

Q 例如：获取群组信息、

权限名称	权限类型	权限状态	可访问的数据范围 配置
获取文件 aily:file:read	应用身份	已开通	-
获取文件 aily:file:read	用户身份	已开通	与用户权限范围一致
上传文件	应用身份	已开通	-

填写信息后，保存

[返回](#)

版本详情

本次发布免审核，提交发布后即可上线使用

应用版本号 *

1.0.0

移动端的默认能力 *

机器人

桌面端的默认能力 *

机器人

*默认的应用功能即用户通过工作台访问应用时打开的功能

更新说明 *

测试使用

应用能力

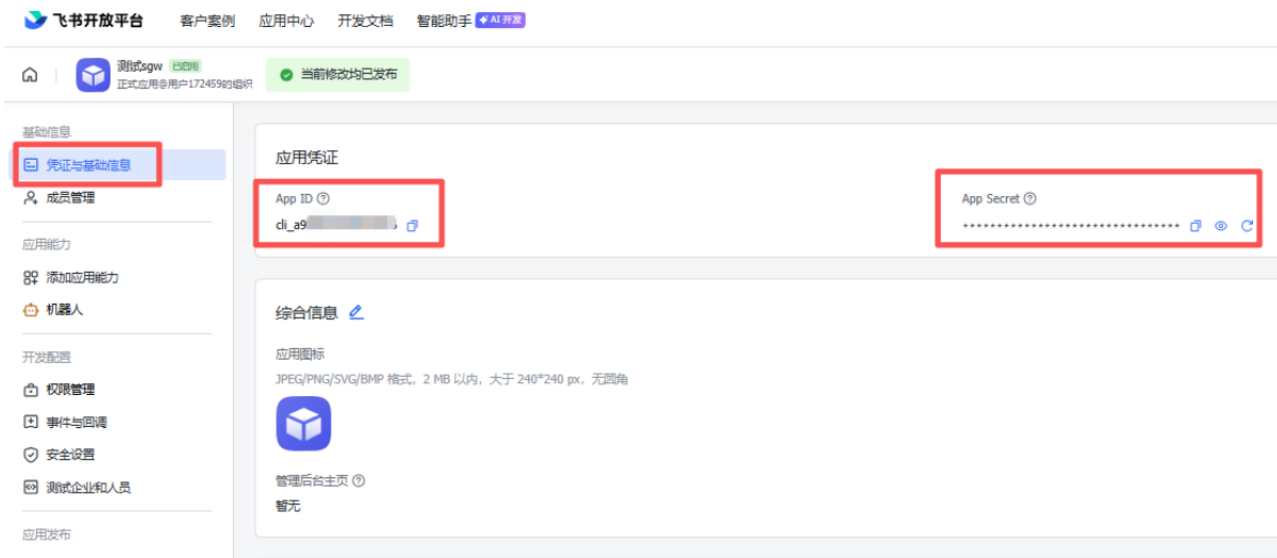
机器人

已启用

发布



点击“凭证与基础信息”，得到AppID和AppSecret的值，将这两值复制出来备用



回到CMD窗口，在这里粘贴上AppID的值后回车



再粘贴上AppSecret的值后回车

```
> 选择一个通道
Feishu (Lark Open Platform)

> 安装 Feishu 插件?
使用本地插件路径

> 选择平台 / Select platform
飞书 (国内版)

> 输入 飞书 App ID (cli_...)
cli_a92810e453b81bc6

* 输入 飞书 App Secret
8DI
```

选择“已完成”回车

```
8DLdnq0LH3m9Zn3h33mCKHLETC3h33h1a

* 选择一个通道
Telegram (Bot API)
WhatsApp (QR link)
Discord (Bot API)
Google Chat (Chat API)
Feishu/Lark (飞书)
Slack (Socket Mode)
Signal (signal-cli)
iMessage (imsg)
> 已完成 (完成)
```

选择yes回车

```
> 技能状态 -----+
符合条件: 51
缺少需求: 0
被允许列表阻止: 0
-----+

* 现在配置技能? (推荐)
> Yes / No
```

选择npm回车

```
o 现在配置技能? (推荐)
  Yes

* 技能安装的首选节点管理器
  > npm
    pnpm
    bun
```

回车后看到如下图

```
o 技能安装的首选节点管理器
  npm

o 钩子 -----+
  钩子让您能够在代理命令发出时自动执行操作。
  例如: 在您发出/new时将会话上下文保存到内存中。

  了解更多: https://docs.clawd.bot/hooks
  -----+

* 启用钩子?
  [•] 暂时跳过
  [ ] [🔗] boot-md
  [ ] [🔗] command-logger
  [ ] [🔗] session-memory
```

需要注意，这里是多选，不能直接回车，直接回车会有一堆问题。我们使用上下箭头，然后敲击space空格键选择boot-md、command-logger、session-memory

```
o 钩子
钩子让您能够在代理命令发出时自动执行操作。
例如：在您发出/new时将会话上下文保存到内存中。

了解更多：https://docs.clawd.bot/hooks

* 启用钩子?
[ ] 暂时跳过
[+] boot-md (Run BOOT.md on gateway startup)
[+] command-logger (Log all command events to a centralized audit file)
[+] session-memory (Save session context to memory when /new command is issued)
```

如上图，变为+后代表选中了，然后敲回车结果如下。

```
o 钩子
钩子让您能够在代理命令发出时自动执行操作。
例如：在您发出/new时将会话上下文保存到内存中。

了解更多：https://docs.clawd.bot/hooks

o 启用钩子?
o boot-md, o command-logger, o session-memory

o 钩子已配置
启用了 3 个钩子: boot-md, command-logger, session-memory

您可以稍后使用以下命令管理钩子:
openclaw-cn hooks list
openclaw-cn hooks enable <name>
openclaw-cn hooks disable <name>

Config overwrite: C:\Users\82634\.openclaw\openclaw.json (sha256 4966c862927747da7a9b3eeae77509eb41
1904a04a, backup=C:\Users\82634\.openclaw\openclaw.json.bak)

* 网关服务运行时
> Node (recommended) (Required for WhatsApp + Telegram. Bun can corrupt memory on reconnect.)
```

继续敲回车

```
0 Installing Gateway service...
Installed Scheduled Task: Openclaw Gateway
Task script: C:\Users\82634\.openclaw\gateway.cmd
o 网关服务已安装。
|
o
Feishu: ok
Agents: main (default)
Heartbeat interval: 30m (main)
Session store (main): C:\Users\82634\.openclaw\agents\main\sessions\sessions.json (0 entries)

o 可选应用
  为额外功能添加节点:
  - macOS应用 (系统+通知)
  - iOS应用 (相机/画布)
  - Android应用 (相机/画布)

o 控制界面
  网页界面: http://127.0.0.1:18789/
  网页界面 (带令牌): http://127.0.0.1:18789/?token=MyGatewayToken
  网关WS: ws://127.0.0.1:18789
  网关: 可访问
  文档: https://docs.clawd.bot/web/control-ui

o 启动TUI (最佳选项!)
  这是定义性的操作, 使您的智能体成为您的。
  请慢慢来。
  您告诉它的越多, 体验就会越好。
  我们将发送: "醒来吧, 我的朋友!"

o 令牌
  网关令牌: 网关+控制界面的共享认证。
  存储在: ~/.openclaw/openclaw.json (gateway.auth.token) 或OPENCLAW_GATEWAY_TOKEN。
  网页界面在此浏览器的localStorage中存储副本 (clawdbot.control.settings.v1)。
  随时获取带令牌的链接: openclaw-cn dashboard --no-open

* 您想如何孵化您的机器人?
  > 在TUI中孵化 (推荐)
    打开网页界面
    稍后执行
```

会新弹出一个窗口

```
clawdbot-gateway x +
09:14:06 [info]: [ 'client ready' ]
09:14:06 [browser/server] Browser control listening on http://127.0.0.1:18791/ (auth=token)
09:14:06 Hook warning: Hook 'boot-md' has no events defined in metadata
09:14:06 Hook warning: Hook 'command-logger' has no events defined in metadata
09:14:06 Hook warning: Hook 'session-memory' has no events defined in metadata
09:14:06 [feishu] starting feishu[default] (mode: websocket)
09:14:06 [gateway] device pairing auto-approved device=d9192c6ad7c3edc1ebea727e5bbe4388a4754aab18d6c227cc95c2c4
role=operator
09:14:06 [feishu] feishu[default]: bot open_id resolved: ou_54dedfba9a547c4ef813d6172765cbc2
09:14:06 [info]: [ 'event-dispatch is ready' ]
09:14:06 [feishu] feishu[default]: starting WebSocket connection...
09:14:06 [info]: [
  '[ws]',
  'receive events or callbacks through persistent connection only available in self-build & Feishu app, Configu
' +
  '
  Developer Console(开发者后台) \n' +
  ->\n' +
  Events and Callbacks(事件与回调)\n' +
  -> \n' +
  Mode of event/callback subscription(订阅方式)\n' +
  -> \n' +
  Receive events/callbacks through persistent connection(使用 长连接 接收事件/回调)'
]
09:14:06 [feishu] feishu[default]: WebSocket client started
09:14:06 [ws] res ✓ health 284ms conn=e3e2d929...3651 id=fcef4e89...2be6
09:14:06 [gateway] update available (latest): v2026.1.24-3 (current v0.1.6). Run: openclaw-cn update
09:14:06 [info]: [ '[ws]', 'ws client ready' ]
09:14:08 [bonjour] gateway name conflict resolved; newName="LAPTOP-EA20G705 (Clawdbot) (2)"
09:14:08 [bonjour] gateway hostname conflict resolved; newHostname="LAPTOP-EA20G705-(2)"
```

此时已经安装成功了。新窗口我们先不管，回到之前的cmd窗口，有如下三个选项

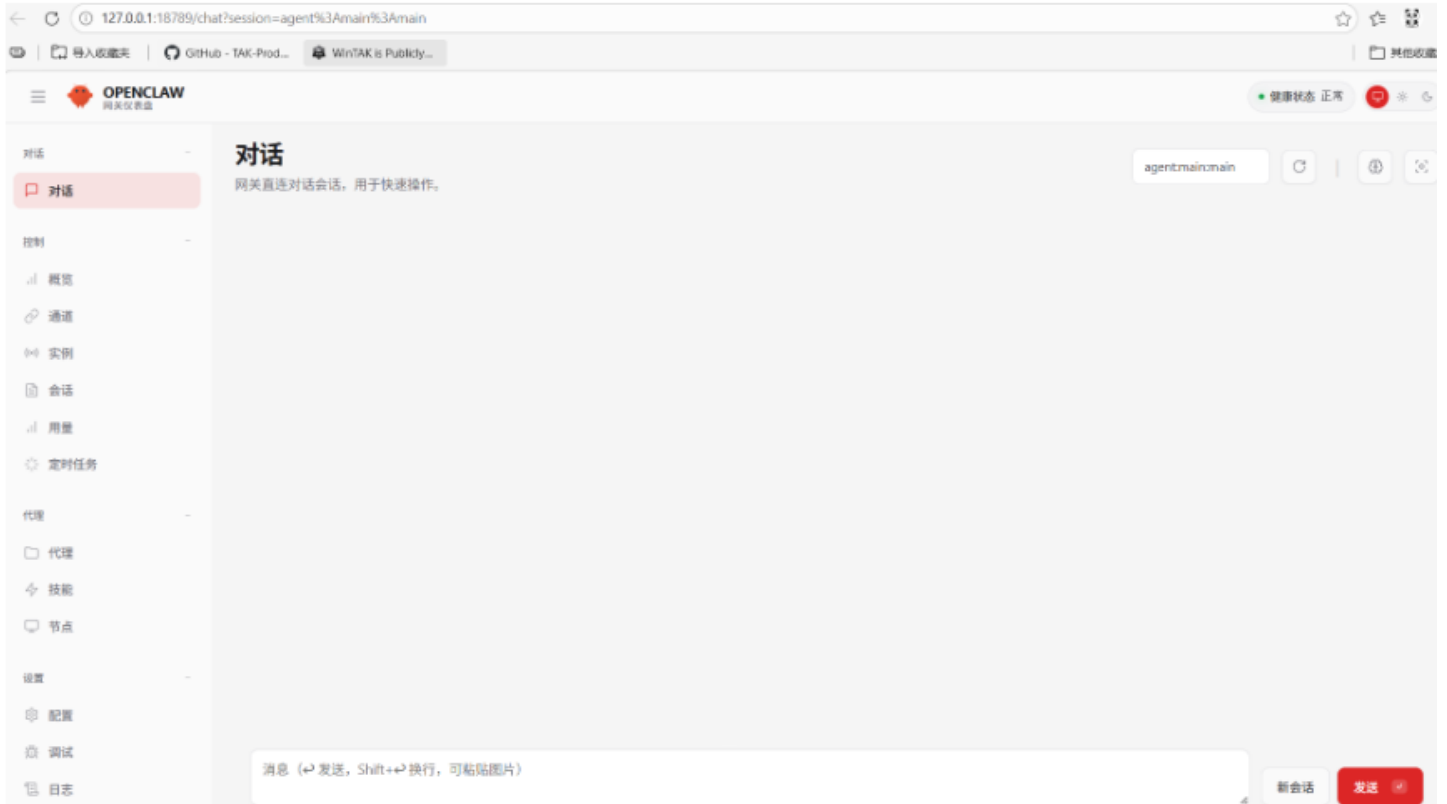
```
控制界面 -----+
网页界面: http://127.0.0.1:18789/
网页界面 (带令牌): http://127.0.0.1:18789/?token=MyGatewayToken
网关WS: ws://127.0.0.1:18789
网关: 可访问
文档: https://docs.clawd.bot/web/control-ui
-----+

启动TUI (最佳选项!) -----+
这是定义性的操作, 使您的智能体成为您的。
请慢慢来。
您告诉它的越多, 体验就会越好。
我们将发送: "醒来吧, 我的朋友!"
-----+

令牌 -----+
网关令牌: 网关+控制界面的共享认证。
存储在: ~/.openclaw/openclaw.json (gateway.auth.token) 或OPENCLAW_GATEWAY_TOKEN
网页界面在此浏览器的localStorage中存储副本 (clawdbot.control.settings.v1)。
随时获取带令牌的链接: openclaw-cn dashboard --no-open
-----+

你想如何孵化你的机器人?
> 在TUI中孵化 (推荐)
  打开网页界面
  稍后执行
```

其中“在TUI中孵化（推荐）”代表接下来在命令行与大模型进行交互，“打开网页界面”代表接下来在浏览器进行交互。我们选择“打开网页界面”，敲回车，会自动打开浏览器。如下



备注：此时本地部署正式完成。想要关闭服务的话，直接关闭cmd窗口即可。

以后启动openclaw

代码块

```
1 openclaw-cn gateway run
```

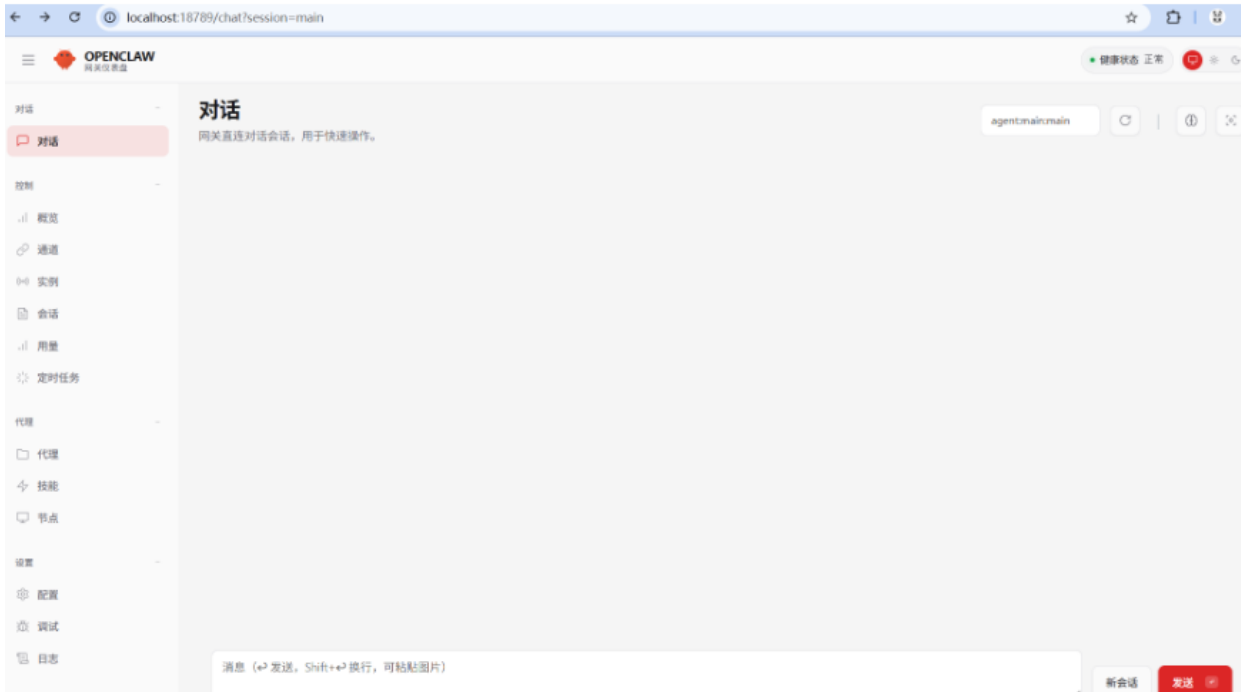
```
OpenClaw-CN 0.1.6 (c0ca28c) - 将"I'll reply later"变为"my bot replied instantly".
8:10:52 [canvas] host mounted at http://127.0.0.1:18789/___clawdbot___/canvas/ (root C:\Users\82634\clawd\canvas)
8:10:52 [heartbeat] started
8:10:52 [gateway] agent model: deepseek/deepseek-chat
8:10:52 [gateway] listening on ws://127.0.0.1:18789 (PID 16164)
8:10:52 [gateway] listening on ws://[::]:18789
8:10:52 [gateway] log file: \tmp\clawdbot\clawdbot-2026-02-28.log
8:10:52 [info]: [ 'client ready' ]
8:10:52 [browser/server] Browser control listening on http://127.0.0.1:18791/ (auth=token)
8:10:52 Hook warning: Hook 'boot-md' has no events defined in metadata
8:10:52 Hook warning: Hook 'command-logger' has no events defined in metadata
8:10:52 Hook warning: Hook 'session-memory' has no events defined in metadata
8:10:52 [feishu] starting feishu[default] (mode: websocket)
8:10:53 [feishu] feishu[default]: bot open_id resolved: ou_f09c6d19f99743754c74b444b6069593
8:10:53 [info]: [ 'event-dispatch is ready' ]
8:10:53 [feishu] feishu[default]: starting WebSocket connection...
8:10:53 [info]: [
  'ws',
  'receive events or callbacks through persistent connection only available in self-build & Feishu app, Configured in:\n
+
  Developer: Feishu (开发者工会) \n
+

```

浏览器访问：<http://localhost:18789/chat?session=main&token=MyGatewayToken>

注意：参数里的MyGatewayToken是你前边设置过的，不是随便写的

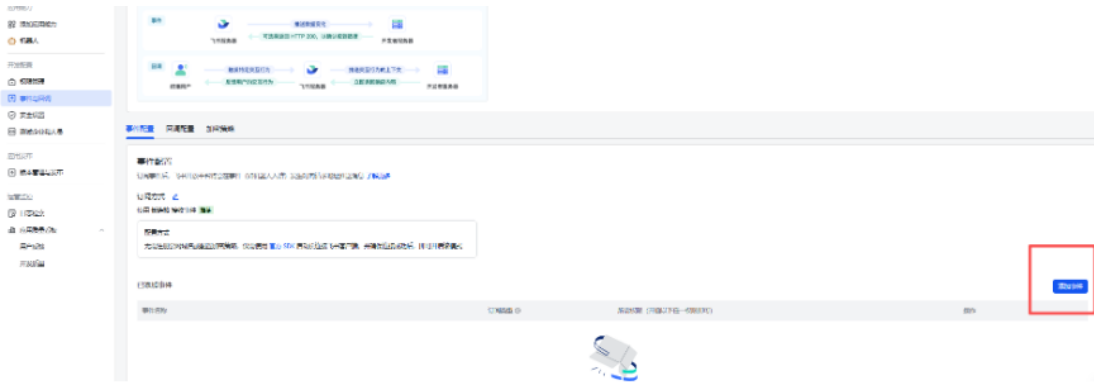
浏览器访问结果如下：



此时，我们只是在本地部署完毕了，但是飞书机器人还不能实时与我们本地openclaw交互，需要在飞书继续做一些配置



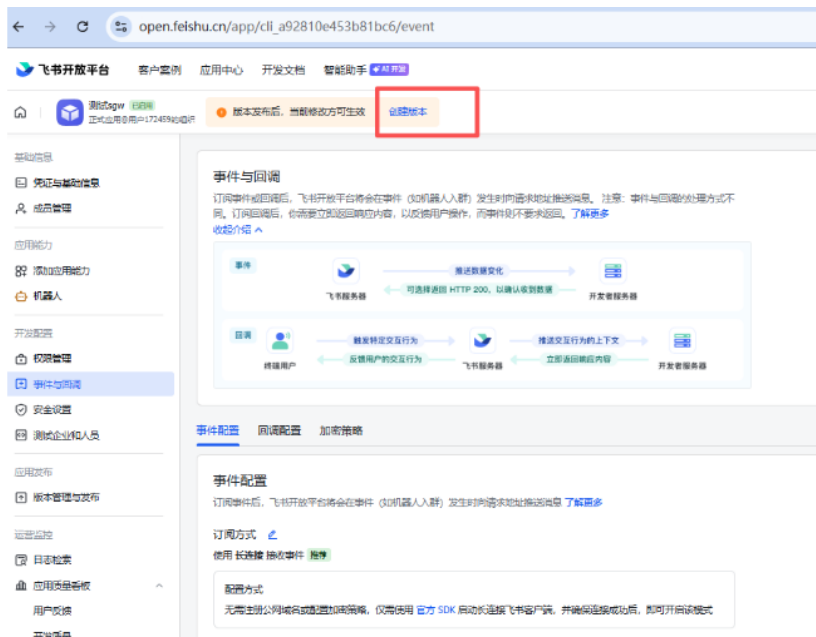
我们继续添加事件，填入官方设置好的事件名



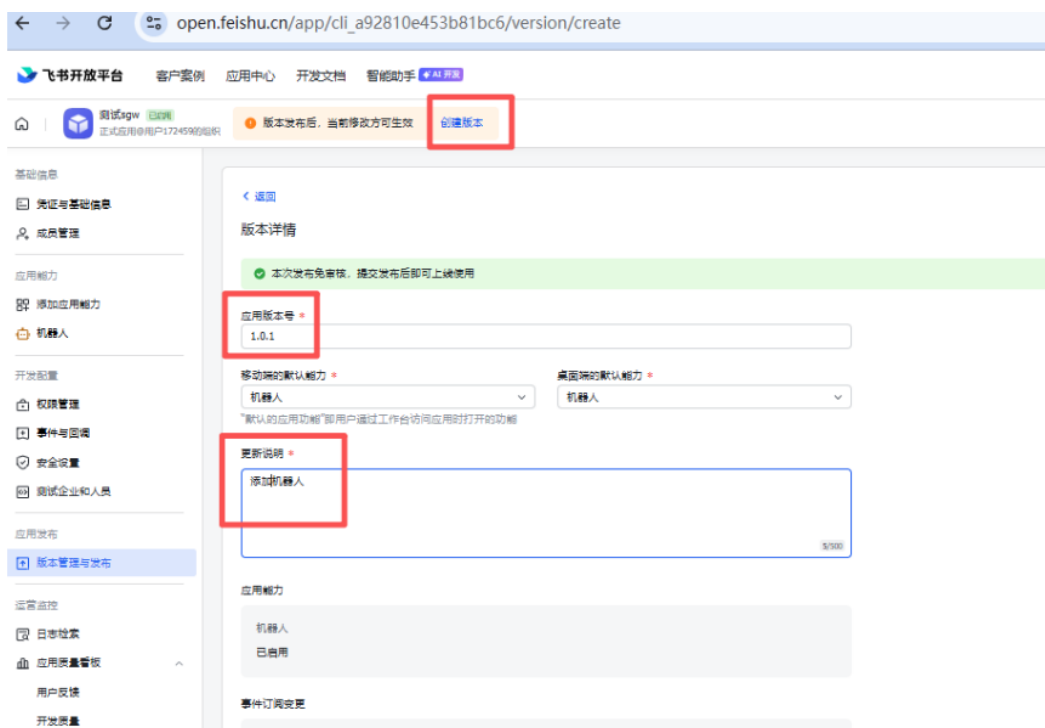
搜索“机器人”，点击添加



保存好之后，我们和之前一样，需要重新发布一个版本



填写内容 (记得保存)



如果以后想让openclaw操作chrome浏览器，那么需要安装浏览器插件

代码块

- 1 # 安装插件
- 2 openclaw-cn browser extension install
- 3 # 查看插件位置
- 4 openclaw-cn browser extension path

```
C:\Users\82634>openclaw browser extension path
Config warnings:\n- plugins.entries.feishu: plugin feishu: duplicate plugin id detected; later plugin may be overr
rs\82634\AppData\Roaming\npm\node_modules\openclaw\extensions\feishu\index.ts)

♥️ OpenClaw 2026.2.26 (bc50708) - It's not "failing," it's "discovering new ways to configure the same thing wrong

o Config warnings -----
- plugins.entries.feishu: plugin feishu: duplicate plugin id detected; later plugin may
  be overridden
  (C:\Users\82634\AppData\Roaming\npm\node_modules\openclaw\extensions\feishu\index.ts)

~\.openclaw\browser\chrome-extension
copied to clipboard.

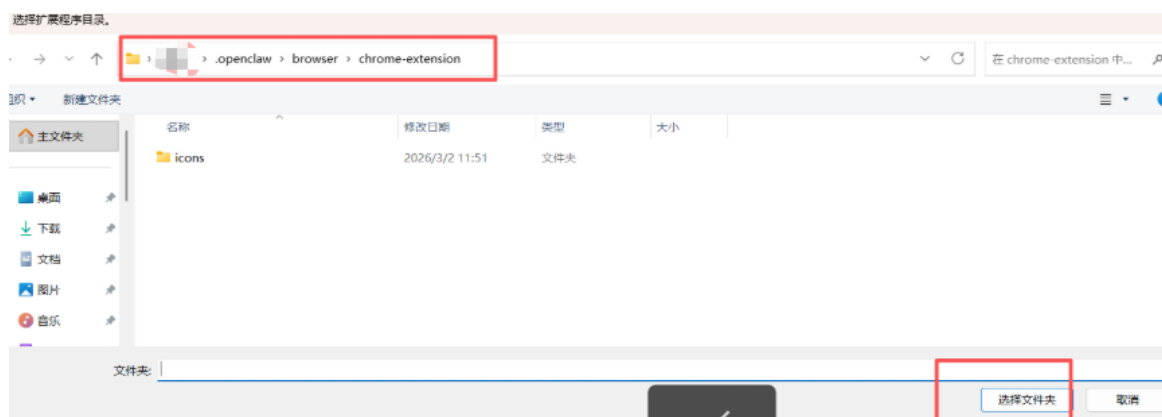
C:\Users\82634>
```

打开Chrome浏览器，在浏览器里输入：

```
代码块
1 chrome://extensions
```



选择插件文件夹



查看结果

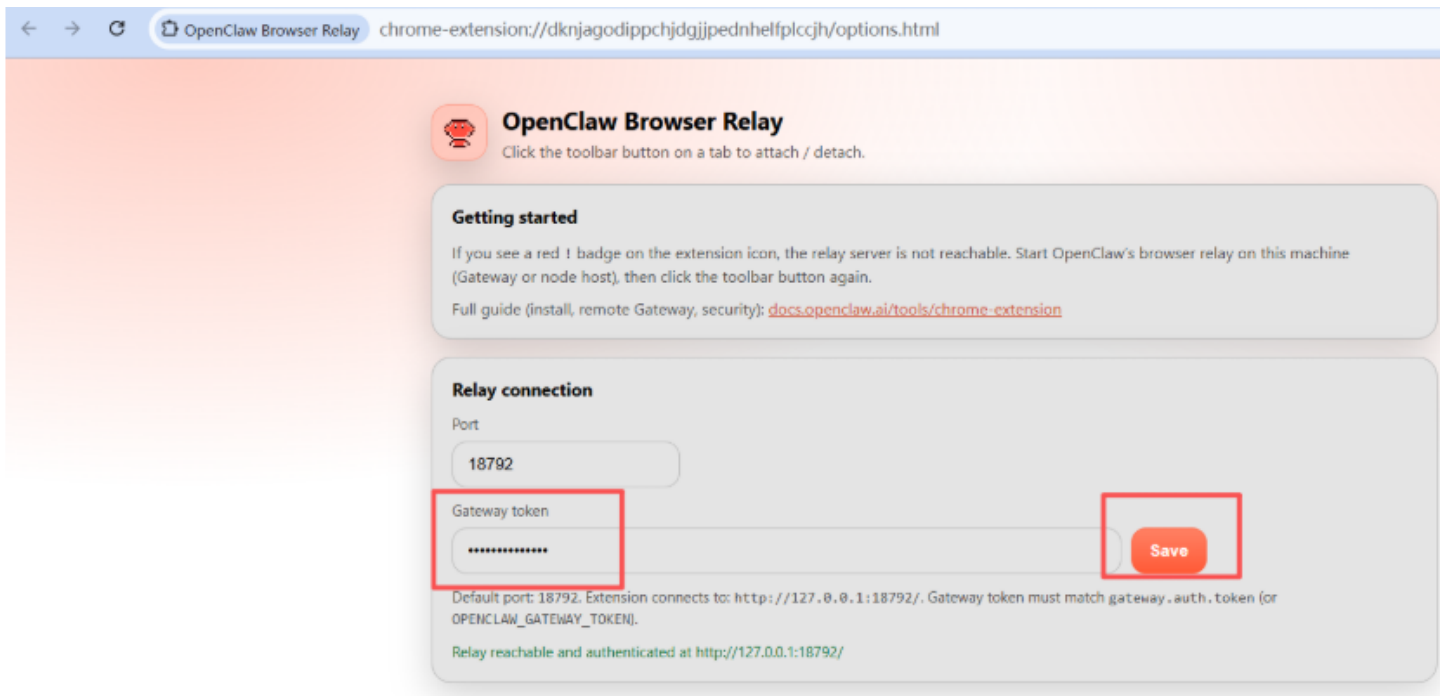
查看结果



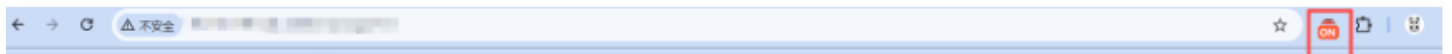
点击刷新按钮



写上之前建好的token，我的是MyGatewayToken



注意，上边这个页面不能关闭，否则openclaw无法控制浏览器。在你想让openclaw控制的页面，打开这个插件（处于on状态），如下



此时，openclaw就能控制你的浏览器当前这个页面了。

停止openclaw的开机自启

以管理员打开cmd，执行如下命令

代码块

```
1 schtasks /Delete /F /TN "OpenClaw Gateway"
```

2.3 常见问题

- Node.js 版本低：升级至 22.0.0 以上。
- 权限不足：加 sudo 或用管理员身份运行终端。
- 模型连接失败：检查 API Key 正确性。

3. 卸载openclaw

如果之前开启了开机启动openclaw，则需要先停服务

代码块

```
1  openclaw-cn gateway stop
2  # 或者尝试
3  openclaw-cn service stop
```

如果没有配置过开机启动的话，直接把cmd网关窗口关闭即可停止服务，然后新开一个cmd窗口执行卸载命令

代码块

```
1  # 卸载openclaw
2  npm uninstall -g openclaw-cn
```

```
C:\Users\82634>npm uninstall -g openclaw-cn
removed 652 packages in 6s
C:\Users\82634>
```

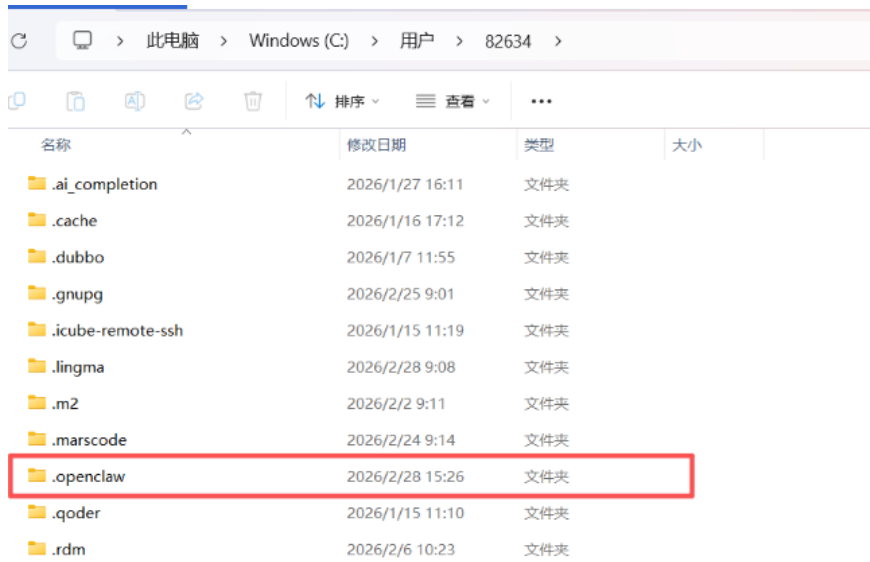
验证卸载是否成功：

代码块

```
1  openclaw-cn --version
```

```
C:\Users\82634>openclaw-cn --version
'openclaw-cn' 不是内部或外部命令，也不是可运行的程序
或批处理文件。
C:\Users\82634>
```

删除系统磁盘里的openclaw文件夹



The image shows a Windows File Explorer window with the address bar displaying the path: 此电脑 > Windows (C:) > 用户 > 82634 >. The window shows a list of folders with columns for Name, Modified Date, Type, and Size. The folder '.openclaw' is highlighted with a red box.

名称	修改日期	类型	大小
.ai_completion	2026/1/27 16:11	文件夹	
.cache	2026/1/16 17:12	文件夹	
.dubbo	2026/1/7 11:55	文件夹	
.gnupg	2026/2/25 9:01	文件夹	
.icube-remote-ssh	2026/1/15 11:19	文件夹	
.lingma	2026/2/28 9:08	文件夹	
.m2	2026/2/2 9:11	文件夹	
.marscode	2026/2/24 9:14	文件夹	
.openclaw	2026/2/28 15:26	文件夹	
.qoder	2026/1/15 11:10	文件夹	
.rdm	2026/2/6 10:23	文件夹	

这将删除所有的对话历史、已配置的 API Key 和钩子设置。如果你以后还想保留这些数据，请不要删除此文件夹，或者将其备份到别处。