如何使用OpenCode桌面端快速开发一个Wordpress插件

疯狂小编 | CrazyEditor | 2026-06-18

使用 OpenCode 桌面端开发 WordPress 插件，最大的优势在于它不仅仅是一个代码生成器，更是一个“可编排的小型工程团队”。为了让你不走弯路，
建议严格遵循“先规划、后执行”的核心原则。以下是详细的实战步骤：

1. 环境准备与项目初始化

- 安装与启动：下载并安装 OpenCode 桌面端（Beta版支持 macOS、Windows 和 Linux）。启动后，在界面中“选择文件夹”，指定你的 WordPress 插件开发目录（例如 wp-content/plugins/my-custom-plugin）。
- 外观配置：根据个人习惯，在左下角的配置按钮中调整主题（如深色主题）和字体，以获得舒适的编码体验。

2. 核心心法：善用 Plan 与 Build 模式

这是避免“AI 乱改代码”导致项目崩溃的最重要机制。OpenCode 通过 Tab 键在两个核心模式间切换：

- □ □ Plan 模式（只读规划）此模式下 AI 不能修改任何代码。它专门用于分析现有代码库、梳理逻辑并建议实现策略。
- □ □ Build 模式（默认执行）此模式下 AI 拥有读写文件、执行 Shell 命令的权限，用于实际开发。

3. WordPress 插件开发实战流程

假设我们要开发一个“自定义文章类型与前端展示”的插件，请按以下流程操作：

第一步：在 Plan 模式下搭建架构（防走弯路关键）

在对话框输入需求，例如：“帮我规划一个 WordPress 插件，需要注册一个名为 ‘Books’ 的自定义文章类型，包含书名、作者
自定义字段，并提供一个短代码 [book_list]
用于在前端展示。请先给出详细的文件结构和实现方案，不要修改任何文件。”

- AI
  的作用：它会为你规划出 plugin.php（主文件）、includes/（类文件）、assets/（样式脚本）等结构，并列出开发步骤。
- 你的动作：审阅方案，确认无误后，按 Tab 键切换到 Build 模式。

第二步：在 Build 模式下逐步实施

在 Build 模式下，将大任务拆解为小指令，避免一次性让 AI 生成过多代码导致上下文混乱：

- 指令示例 1：“根据刚才的计划，创建插件主文件 plugin.php，添加标准的 WordPress 插件头注释，并引入 includes 目录下的类。”
- 指令示例 2：“在 includes 目录下创建 CustomPostType.php 类，实现 Books 文章类型的注册。”
- 指令示例 3：“编写 Shortcode.php 类，实现 [book_list] 的数据库查询和 HTML 渲染。”

第三步：利用 @ 符号提供精准上下文

不要让 AI 在整个项目里“瞎猜”。当你在修改某个功能时，使用 @ 符号将相关文件拉入上下文：

- 正确做法：“当前的查询逻辑有性能问题，请帮我优化 @includes/Shortcode.php，使用 WP_Query 的缓存机制。”
- 错误做法：“修复查询 bug。”（缺乏上下文，AI 容易改错文件）。

第四步：调试与容错

- 撤销机制：如果 AI 在 Build 模式下把代码改坏了，不要慌，直接在终端输入 /undo 即可撤销上一次的更改，输入 /redo 恢复。
- 测试驱动：你可以让 AI 帮你编写 PHPUnit 测试，或者让它执行 Shell 命令来检查 PHP 语法错误。

□ □ 进阶提效技巧

1. 具体描述需求：不要说“让它更好”，而是说“用 async/await 重构 JS 文件并添加错误处理”；不要说“修复 bug”，而是说“点击提交时 auth.ts 第 45 行报 undefined 错误”。
2. 利用免费模型：OpenCode 桌面端内置了如 GPT-5

Nano、GLM4.7 等限时免费模型，你可以零成本进行前期的架构规划和代码生成。

3. 支持拖拽截图：如果你有 Figma 设计稿或 UI 截图，可以直接拖拽到 OpenCode 终端中，它能理解图片并据此生成对应的前端 CSS/HTML 代码。

总结一下：通过 “Plan 模式定方向 -> Build 模式做执行 -> @ 符号给上下文 -> /undo 兜底” 的闭环，你可以最大程度地发挥 OpenCode 桌面端的 Agent 能力，高效且安全地完成 WordPress 插件开发。