

DeepSeek Harness

从入门到精通

系统讲解：产品 · 用法 · 案例 · 同行对比



联合出品：



公众号：技术领导力



公众号：AI新猿人

本材料面向技术爱好者、想系统上手DeepSeek Harness的读者，共六章，从概念到实战再到选型，逐层深入。

章节	主题	你将掌握
第 1 章 入门篇	认识 DeepSeek Harness	定位、背景、核心公式、插件理念
第 2 章 产品篇	功能与架构拆解	入口模式、计划/代码模式、工具、沙盒、插件
第 3 章 用法篇	从入门到上手	Web UI、CLI、Python SDK、写插件
第 4 章 案例篇	真实场景实战	开发重构、数据分析、办公自动化、多 Agent
第 5 章 对比篇	与四大竞品对比	WorkBuddy / Trae Work / Qoder Work / Codex

阅读建议：想快速上手直接看第 3 章；想选型对比直接看第 5 章；想彻底搞懂架构再看第 2 章。

第 1 章

入门篇

认识 DeepSeek Harness —— 它是什么、为什么出现、核心思想

《DeepSeek Harness 从入门到精通》

什么是 DeepSeek Harness

1-1



关键区分：Harness 是「框架/引擎」，不是聊天机器人，也不是某个具体的 AI 应用——它是用来「组装 Agent」的开源底座。

核心公式：Model + Harness = Agent

这是理解 Harness 最重要的一句话：模型负责「思考」，Harness 负责「执行」，两者相加才是一个完整的智能体。

分工	谁负责	职责
Model（模型）	DeepSeek V4-Pro / V4-Flash 等	推理、规划、生成回复，决定「做什么」
Harness（框架）	dsh + Cordis 插件	工具调用、文件读写、命令执行、编排，决定「怎么做」
Agent（智能体）	两者组合	能自主完成任务的完整执行体

一句话记忆：换模型就像换发动机，换插件就像换工具头——发动机和工具头可以独立升级，这就是「可组合」的含义。



理念价值：把「Agent 能力」变成可安装、可分发、可替换的零件，从而形成开源插件生态（npm 分发、dsh-plugin 话题）。

第 2 章

产品篇

功能与架构拆解 —— 入口模式、计划/代码模式、核心工具、沙盒与插件

《DeepSeek Harness 从入门到精通》

产品形态总览：四种使用入口

Harness 不是单一应用，而是一套「框架 + 多个入口」，用户可按需选择交互方式。

入口	形态	适用场景
Web UI	图形界面，dsh web 启动	日常交互、观察 agent 执行、审批操作
CLI (Headless)	命令行一次性任务	脚本化、CI/CD、批量任务
Python SDK	程序化调用 deepseek_harness	嵌入自己的程序/服务
插件开发	TypeScript 编写插件	扩展 Harness 自身能力

Web UI 用来「看」，CLI 用来「跑批」，SDK 用来「嵌」，插件用来「扩」。

CLI 入口模式: web / headless / plugin

2-2

命令	用途
dsh --profile <name>	启动位于 \$DSH_HOME/profiles/<name> 的指定 profile
dsh --profile headless "任务"	运行一次全新持久化会话，打印最终答案后退出
dsh web	--profile web 的别名，启动 Web UI
dsh plugin --profile <name> <pnpm 参数>	管理该 profile 的插件（转发给 pnpm）

```
npx @deepseek-ai/dsh web
dsh --profile headless "run the tests"
dsh plugin --profile web add <plugin>
```

计划模式 (Plan Mode)：先想清楚再动手

2-3

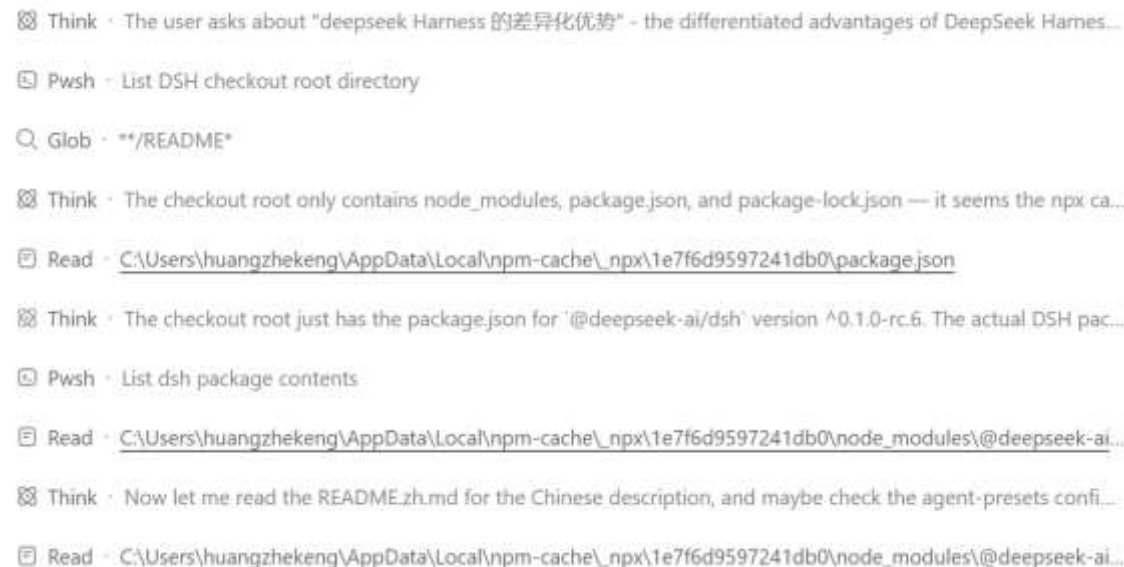


适用场景：复杂任务、多步骤改动前，先让 agent 出计划给你审，避免「直接动手改坏」。

「调用模型 → 运行工具 → 重复」。它把这个循环做成一个极小的、可替换的驱动器（Agent Loop）

其余一切（沙箱、权限、压缩、重试、子代理...）都以插件 + 事件的方式挂在外围

而「代码模式（Code Mode）」则把「运行工具」升级为「让模型写一段程序，在隔离环境里批量调用工具」。

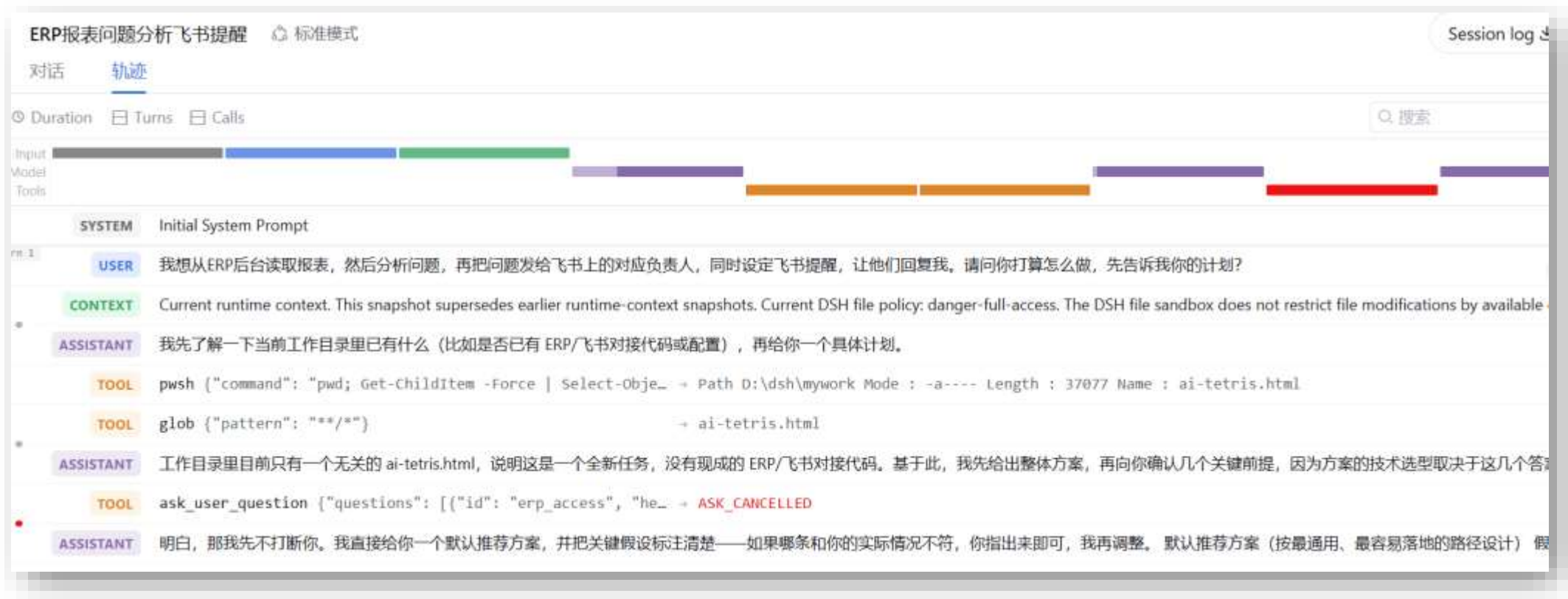


```
⊗ Think · The user asks about "deepseek Harness 的差异化优势" - the differentiated advantages of DeepSeek Harnes...  
📁 Pwsh · List DSH checkout root directory  
🔍 Glob · **/README*  
⊗ Think · The checkout root only contains node_modules, package.json, and package-lock.json — it seems the npx ca...  
📄 Read · C:\Users\huangzhekeng\AppData\Local\npm-cache\npx\1e7f6d9597241db0\package.json  
⊗ Think · The checkout root just has the package.json for '@deepseek-ai/dsh' version ^0.1.0-rc.6. The actual DSH pac...  
📁 Pwsh · List dsh package contents  
📄 Read · C:\Users\huangzhekeng\AppData\Local\npm-cache\npx\1e7f6d9597241db0\node_modules\@deepseek-ai...  
⊗ Think · Now let me read the README.zh.md for the Chinese description, and maybe check the agent-presets confi...  
📄 Read · C:\Users\huangzhekeng\AppData\Local\npm-cache\npx\1e7f6d9597241db0\node_modules\@deepseek-ai...
```

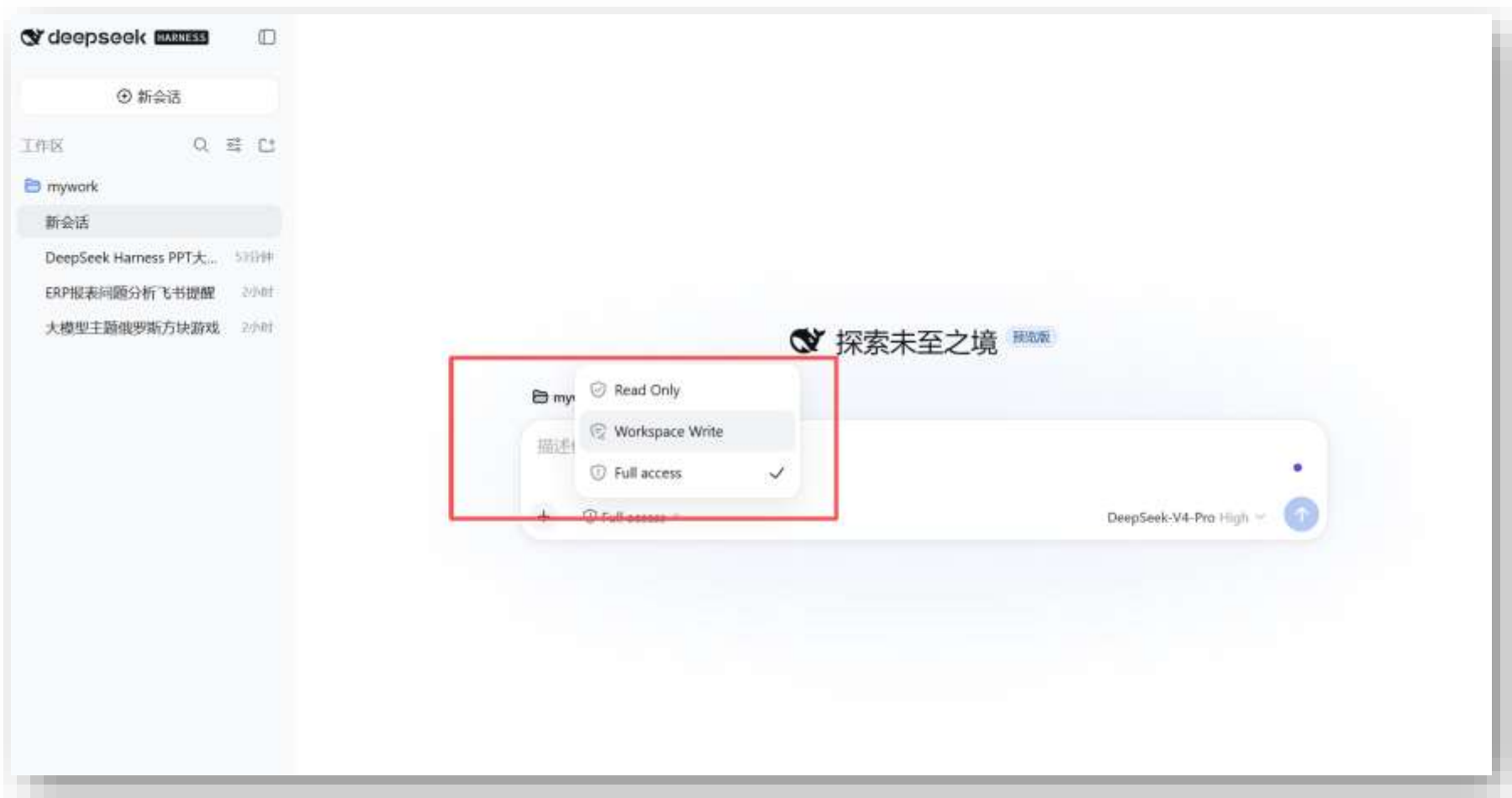
关键点：模型看到的每一段内容都来自会话日志（「模型可见即已记录」），保证可回放、可审计、可恢复。

能力类别	工具 / 机制	作用
执行环境	bash / pwsh	运行 shell 命令，读写文件系统
文件操作	read / write / edit / glob / grep	读写文本、按模式查找、内容检索
任务委派	subagent / subagent_fork	把独立任务交给子智能体
编排	workflow	脚本化大规模编排多个 subagent
目标管理	create_goal / get_goal / update_goal	跨轮次推进长期目标
后台任务	job_list / job_output / job_kill	管理后台命令与子任务
持久终端	terminal_open / send / read...	跨调用保留状态的 shell 会话
技能	skill	加载可复用技能说明

这些工具同样是插件：需要时挂载，不需要时卸载，从而实现「按需装配」的 Agent。



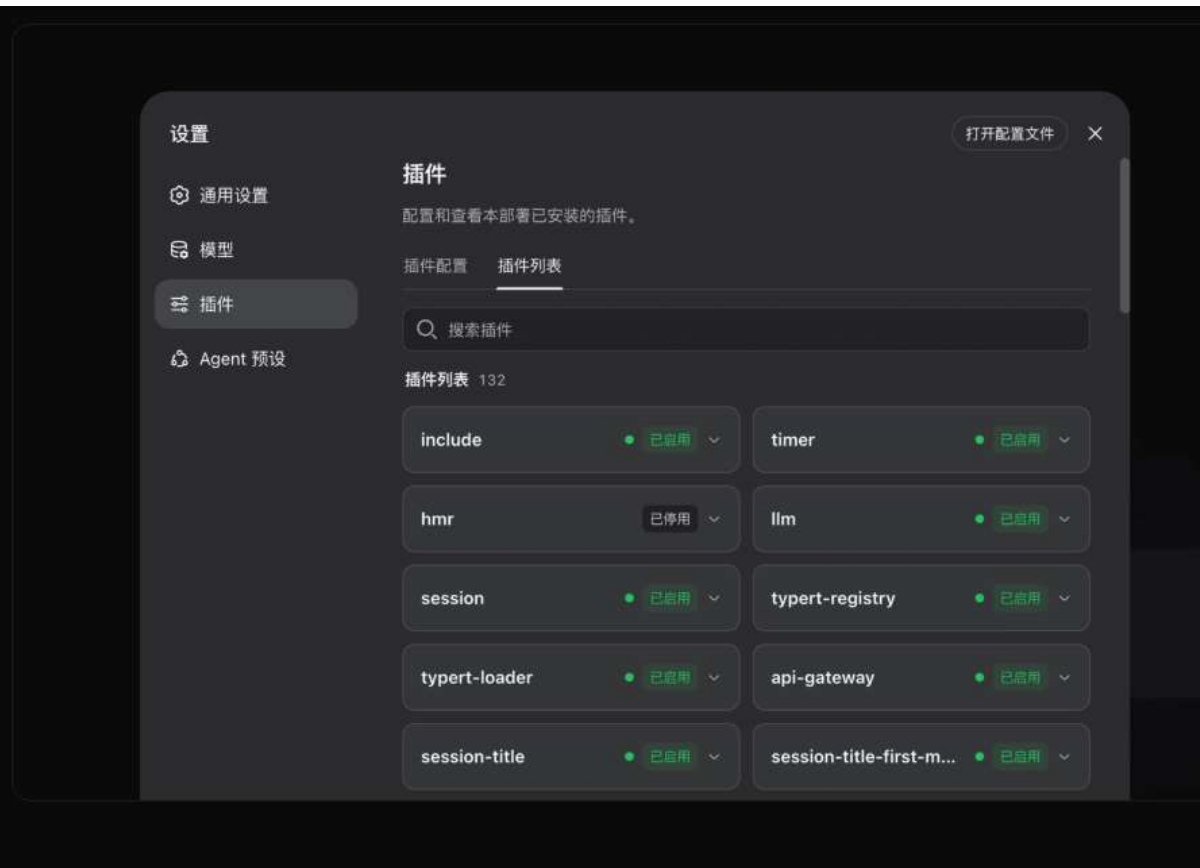
设计含义：把「对话历史」和「执行痕迹」统一成一条可重放的事件流，是 Harness 可审计、可恢复、可 fork 的关键。



核心原则：让 agent 有「能干活的权限」，但把「越界的权限」交给审批和沙盒守住。

一切皆插件

DeepSeek Harness 基于 Cordis 插件系统构建。模型、工具、技能、会话、沙箱、存储、循环、调度、UI 等所有 Agent 能力均由插件提供，并通过 Cordis 服务与事件彼此协作。开发者无需改动 DeepSeek Harness 源码，即可在配置层选择、替换或扩展任一能力。



把 Harness 拆成可插拔的「服务 + 事件 + 接缝」，让每一层都能被替换。

模型		百万tokens输入（缓存命中）	百万tokens输入（缓存未命中）	百万tokens输出
deepseek-v4-flash	空闲时段	0.05元	1.5元	4.5元
	高峰时段	0.10元	3.0元	9.0元
deepseek-v4-pro	空闲时段	0.15元	4.5元	13.5元
	高峰时段	0.30元	9.0元	27.0元

成本：框架本身 0 元，真正的账在「模型 API」和「自托管运维」两处。

第 3 章

用法篇

从入门到上手 —— Web UI、CLI、Python SDK、写插件、排错

《DeepSeek Harness 从入门到精通》

用途	前置要求
Web UI / CLI	安装 Node.js, 然后 npx @deepseek-ai/dsh web (无需预装 Harness)
从源码运行	Git + pnpm: git clone → pnpm install → pnpm run build → pnpm dsh web
Python SDK	Python 3.10+、Git; 平台 Linux x64/arm64 或 macOS 14+ arm64
凭据	DeepSeek 兼容的 API 端点与密钥 (DEEPSEEK_API_KEY)
工作区	一个 agent 可修改的隔离 workspace 目录
<pre>export DEEPSEEK_API_KEY=sk-your-key-here # 可选: OpenAI 兼容代理 / 模型 / 系统提示词 # export DEEPSEEK_BASE_URL=http://127.0.0.1:8000/v1 # export DSH_MODEL=deepseek-v4-flash</pre>	

提示: Windows 桌面用户主要用 Web UI; Python SDK 官方组合目前不支持 Windows agent (依赖 POSIX 终端)。

5 分钟第一个 Agent (Web UI)

3-2



```
npx @deepseek-ai/dsh web  
# 打开浏览器访问 http://127.0.0.1:3080
```

无界面的「一次性任务」入口 —— 一条命令、一个任务、一个持久化会话、一个最终答案。

核心特性

- **一次性执行**: 提交任务 → 等待 Agent 完全停稳 → 打印最终答案 → 退出
- **极简装配**: 直接叠加在 dsh-base 上, 只保留编码 persona、工具模式与 Code Mode worker
- **零服务开销**: 不挂载 Host / HTTP 服务器 / Web 运行时 / 浏览器插件, 不监听任何端口
- **持久化会话**: 每次运行创建全新的持久化 Agent, 天然支持后续续写
- **干净的 I/O**: 最终答案写 stdout; 出错时 code + message 写 stderr; 成功时 stderr 为空
- **可靠的退出码**: turn/end 完成 → 0, 否则 → 1, 便于脚本化 / CI 集成

```
dsh --profile headless "run the tests"
dsh --profile headless "Inspect the repository and fix the failing tests."
dsh --profile web --port 8080 # --port 属于 web 应用
```



Think · The user wants to replace the model names/text on the blocks with their corresponding logo icons. So I need...

明白! 我会把方块上的文字标签替换成各品牌的 Logo 图标 (用 Canvas 矢量绘制的风格化品牌标识, 自包含、无需外部图片)。让我先精确查看当前相关代码段。

Read · ai-tetris.html

Think · Let me find the exact MODEL5 block and drawCell block lines.

Grep · label.标签自动缩放pg-dot(function) buildLegend(const MODEL5

Read · ai-tetris.html

Think · Now I have the exact content. Let me make all edits.

Now I'll make the edits. First, update the MODEL5 array to add logo keys and drop the now-unused text/label fields.

当运行模式为 `code / both`，且代码运行时 `runtime.language = 'python'` 时，DSH 从已注册工具 Schema 确定性生成一套类型化的 Python SDK（dsh-tools 的 `py-types` 渲染器），模型通过它程序化地调用工具。

```
from deepseek_harness import DeepSeekHarness
with DeepSeekHarness(provider="deepseek-official", model="deepseek-v4-flash",
                      cwd=str(workspace), session_root=str(sessions), cordis=str(config)) as h:
    result = h.run("Inspect the repository and fix the failing tests.", session_id="ex-001")
print(result.final_response)
```

适用：把 Agent 嵌入自己的 Python 服务/工具链，复用同一套 API。

- DSH 基于 Cordis 插件框架（TypeScript 依赖注入 / 服务 / 生命周期）
- 配置 = 分层的 YAML patch 叠加成一个「配置树」
- 一切皆插件（Plugin），配置只是「按顺序加载哪些插件 + 传什么参数」

```
- id: tool-bash          # 唯一 id
  name: '@deepseek-ai/dsh-tool-bash' # npm 包名
  config: { timeoutMs: 300000 }      # 按插件 Schema 校验的参数
  disabled: false                  # 可开关

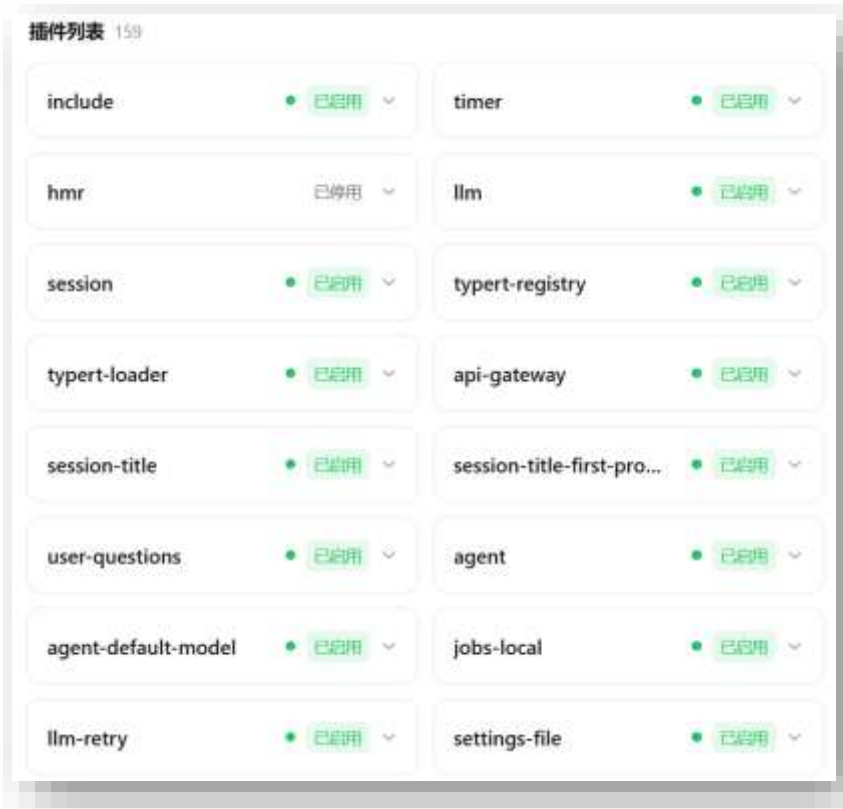
- id: planning
  name: cordis:group              # 分组 + 隔离域
  group: true
  isolate: { planMode: true }
```



关键：模型可换正是 Harness 的核心卖点之一——配置一次提供方，即可在多家模型间切换。

- 一条命令：dsh plugin --profile <name> <pnpm 参数>（转发给 pnpm）
- 插件以 npm 包形式装进 profile 的 node_modules
- 解析顺序：先查 dsh 内置 bundles，再查 profile 自己的 node_modules
- web / headless profile 首次自动初始化；自定义 profile 用 dsh plugin 创建
- Web GUI 提供「插件清单 / 设置」界面查看启用状态与生命周期

```
import type { Context } from '@deepseek-ai/cordis'
export const name = 'my-tool-plugin'
export const inject = ['tools']
export function apply(ctx: Context) {
  ctx.tools.register(/* ... */) // ctx.tools 已就绪
}
```



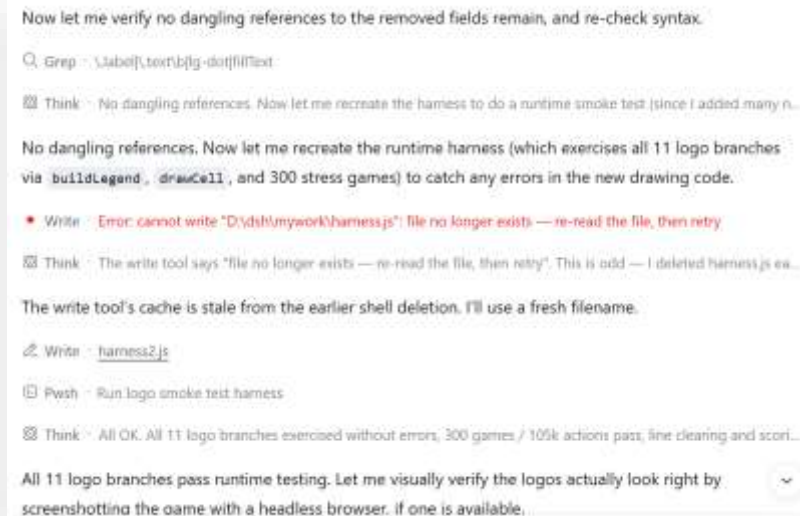
挂载：写入 cordis.yml 并用 --patch 加载：pnpm dsh web --patch ./scratch-plugin/cordis.yml。

工具使用

- 读文件用 read，不要用 cat；搜索用 grep/glob，不要用 shell 的 find/rg
- 新建/整体覆盖用 write，小改动用 edit（更精准）
- 多步任务先列 todo_write 清单，随时更新状态

开发联调

- 确认在正确地址操作：本 GUI 是
- `http://127.0.0.1:3080`
- 改完 Web 产物后要刷新页面验证；客户端插件 HMR 需 `pnpm run dev:web` 运行中
- Web shell 依赖 dsh web 注入 `window.__DSH_BOOT__`，不要另起服务器替代



Now let me verify no dangling references to the removed fields remain, and re-check syntax.

Q: Grep - \label\text\b\g-dot\fillText

Think: No dangling references. Now let me recreate the harness to do a runtime smoke test (since I added many n...

No dangling references. Now let me recreate the runtime harness (which exercises all 11 logo branches via `buildLegend`, `drawCell`, and 300 stress games) to catch any errors in the new drawing code.

Write: Error: cannot write "D:\dsh\mywork\harness.js": file no longer exists — re-read the file, then retry

Think: The write tool says "file no longer exists — re-read the file, then retry". This is odd — I deleted harness.js ea...

The write tool's cache is stale from the earlier shell deletion. I'll use a fresh filename.

Write: harness2.js

Push: Run logo smoke test harness

Think: All OK. All 11 logo branches exercised without errors, 300 games / 105k actions pass, line clearing and scor...

All 11 logo branches pass runtime testing. Let me visually verify the logos actually look right by screenshotting the game with a headless browser, if one is available.

一条心法：先在小 workspace + 审批模式下验证，再逐步放开权限。

第 4 章

案例篇

真实场景实战 —— 开发重构、数据分析、办公自动化、多 Agent 编排

《DeepSeek Harness 从入门到精通》

案例一：代码开发与重构

4-1

开发“AI大模型俄罗斯方块”小游戏



案例二：数据处理与分析

4-2

多模态读取：文本、CSV、JSON、图片等本地数据批量读取（read / read_image）

精准检索：正则全文搜索（grep）+ 模式匹配定位（glob），快速锁定目标数据

计算执行：PowerShell 脚本执行数据清洗、统计、转换与文件处理（pwsh）

并行分析：Subagent 子任务并行 + Workflow 大规模多智能体编排，扇出加速

联网检索：web_search 实时获取外部/最新数据并溯源引用

长任务自动化：Goal 目标驱动 + 后台任务管理，自动跨轮次完成分析闭环

主要特点

- 工具化组合：能力封装为可组合工具，按需调用
- 并行高效：多子代理并发，显著缩短批量分析周期
- 结果可溯源：文件级操作、搜索来源均可追溯（URL 引用）
- 可控安全：沙箱模式约束文件读写与命令执行，风险可控

二、六维评分对比（短板一目了然）

维度	平均分	低分(≤2)人数	首要改进提及
产品功能	4.28	0	2
品牌形象	4.25	0	0
价格	3.63	0	1
问题反馈处理	3.63	0	1
客服质量	2.81	11	10
客服响应	2.63	14	16

核心洞察：客服是唯一拖后腿的环节。产品、品牌表现优秀（4.2+），但「客服响应 + 客服质量」两项均分不足 3 分，合计被 26 位客户列为首要改进项，占全部样本的 81%。客服体验正在抵消产品优势。

分析目录下的“客户满意度数据”，整理洞察，改进建议

上下文注入 · @deepseek-ai/dsh-system-prompt

Think · The user wants me to analyze “客户满意度数据” (customer satisfaction data) in the directory, organize insig...

Pwsh · List all files in working directory

Think · The relevant file is 'D:\dsh\mywork\上周客户满意度数据.xlsx'. Let me analyze it. There's also a ppt-内容 fold...

任务：调研一个主题并生成PPT：

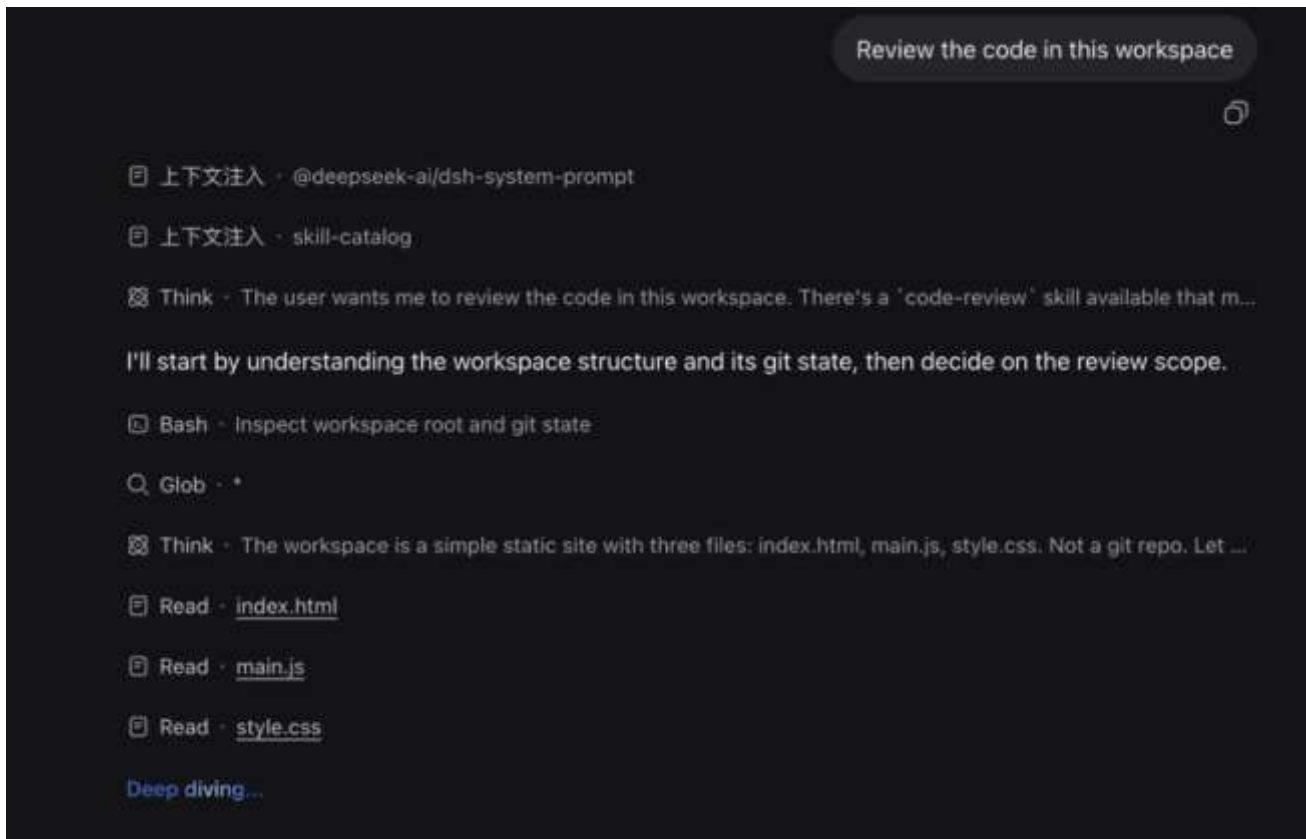
- 1、明确主题：确定范围、受众和页数
- 2、搜索资料：多轮检索，记录来源和数据
- 3、归纳主线：提炼观点，梳理逻辑结构
- 4、搭建大纲：确定每页核心内容
- 5、生成 PPT：脚本自动排版生成 .pptx
- 6、检查交付：核对内容，标注待补充项



案例四：多 Agent 编排 (subagent / workflow)

4-4

- 任务拆解：把一个复杂目标拆成多个独立子任务
- 并行分工：多个 Agent 同时处理不同子任务
- 结果汇总：收集各 Agent 输出，统一整合
- 协同校验：必要时交叉检查、修正冲突
- 形成成果：汇聚为一份完整交付物



The screenshot displays a dark-themed interface for a multi-agent workflow. At the top right, a button reads "Review the code in this workspace". Below it, a list of agents and their actions is shown:

- [-] 上下文注入 · @deepseek-ai/dsh-system-prompt
- [-] 上下文注入 · skill-catalog
- [x] Think · The user wants me to review the code in this workspace. There's a 'code-review' skill available that m...
- I'll start by understanding the workspace structure and its git state, then decide on the review scope.
- [x] Bash · Inspect workspace root and git state
- [x] Glob · *
- [x] Think · The workspace is a simple static site with three files: index.html, main.js, style.css. Not a git repo. Let ...
- [x] Read · [index.html](#)
- [x] Read · [main.js](#)
- [x] Read · [style.css](#)
- Deep diving...

第 5 章

对比篇

与四大竞品对比 —— WorkBuddy · Trae Work · Qoder Work · OpenAI Codex

《DeepSeek Harness 从入门到精通》

DeepSeek Harness 不是孤立出现的产品，它处在 2025–2026 年「AI Agent / 编程智能体」爆发的大赛道里。本章把它与四款代表性产品并排对比。

产品	厂商	一句话定位
DeepSeek Harness	深度求索	开源 Agent 框架，核心口号「一切皆插件」（Everything is a Plugin）
WorkBuddy	腾讯	桌面端 AI 智能体 / 办公工作台，强绑定腾讯生态
Trae Work	字节跳动	AI 原生 IDE，主打「AI 主导开发全流程」，前身 TRAE Solo
Qoder Work	阿里云	全栈 AI 编辑器 / 编程智能体，原「通义灵码」，主打本地优先
OpenAI Codex	OpenAI	闭源云端编程智能体 + 命令行，锁 GPT-5 Codex 系列

结论：Harness 走「卖框架、开放插件、模型可换」路线，四款竞品走「卖产品、绑生态、模型内置」路线。

vs WorkBuddy（腾讯）：办公工作台 vs 开源框架

5-2

WorkBuddy 是什么

是腾讯推出的一款 AI 数字员工 / 桌面智能体（Agent）产品，能替用户「动手干活」：理解自然语言指令后，自动操作电脑上的应用、完成办公任务和重复性工作，帮助提升效率。



与 Harness 的核心差异

维度	WorkBuddy	DeepSeek Harness
产品形态	封闭桌面应用，开箱即用	开源框架 / CLI / SDK，需配置
主要战场	办公自动化 + 腾讯生态	编程、Agent 编排、二次开发
模型自由度	官方集成若干模型	可自由替换任意模型
可扩展性	官方功能为主	一切皆插件，可自写插件
开源	否	是
数据/部署	依赖腾讯云服务	可本地运行 / 自托管

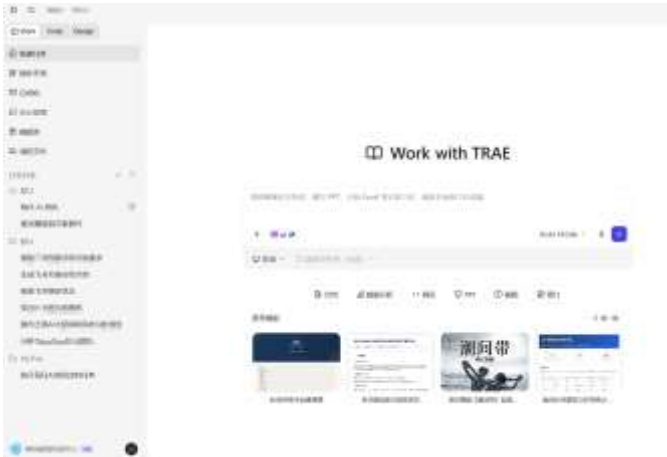
结论：WorkBuddy 适合「不想折腾、重度使用腾讯办公生态」的用户；Harness 适合「要可控、可定制、可换模型、可二次开发」的开发者或团队——「买产品」与「用框架」两条路线。

vs Trae Work（字节）：AI 原生 IDE vs 框架

5-3

Trae Work 是什么

Trae Work 是字节跳动推出的一款 AI 原生办公工作台（超级工作台），定位是帮用户把日常杂活交给 AI 自动完成。核心能力包括自动生成 PPT、数据分析、文档撰写等



与 Harness 的核心差异

维度	Trae Work	DeepSeek Harness
产品形态	完整 IDE（自带编辑器）	框架 / CLI，可嵌入任意工作流
主要战场	应用开发、页面生成、部署	Agent 编排、插件化执行
模型自由度	内置字节系及合作模型	可自由替换任意模型
可扩展性	官方 IDE 插件生态	一切皆插件，可自写插件
开源	否	是
上手门槛	低（装好即用）	中（需了解 CLI / 配置）

结论：Trae Work 是「带着 AI 的完整开发环境」，适合想要全能 IDE 直接开干的开发者；Harness 是「不绑定 IDE 的执行引擎」，适合已有工具链、想把 Agent 能力嵌入现有流程的团队。

vs Qoder Work（阿里）：本地优先 Agent vs 开源框架

5-4

Qoder Work 是什么

Qoder Work 是阿里巴巴推出的一款 AI 办公智能体（桌面 Agent），用户用自然语言下达指令，它就能在电脑上「动手干杂活」——自动操作软件、完成文档、表格、PPT、数据整理等办公任务。



与 Harness 的核心差异

维度	Qoder Work	DeepSeek Harness
产品形态	商业编辑器 / 智能体产品	开源框架 / CLI / SDK
核心卖点	本地优先、记忆/会话/技能引擎	一切皆插件、模型可换
模型自由度	内置通义等模型	可自由替换任意模型
可扩展性	自定义 Skills（受产品约束）	插件 + 二次开发（完全开放）
开源	否	是
数据/部署	本地优先，兼顾云端	可本地 / 自托管

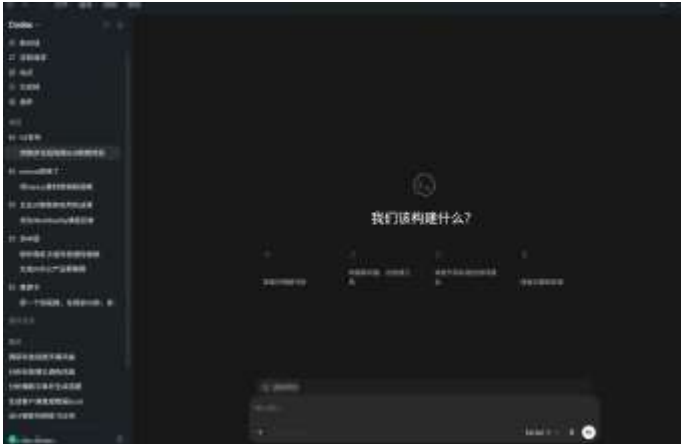
结论：两者在「技能/插件可扩展」上有交集，但 Qoder 是「带本地优先卖点的商业产品」，Harness 是「完全开放的开源框架」。要本地安全 + 现成产品选 Qoder；要深度定制 + 完全可控选 Harness。

vs OpenAI Codex: 闭源云端 vs 开源框架

5-5

Codex 是什么

是 OpenAI 推出的 AI 编程智能体（自主编程代理），不是一个只会聊天的对话框，而是能「动手」写代码、运行任务、自主完成软件工程的 Agent。已经跟 ChatGPT App 合并了，有桌面端、移动端



与 Harness 的核心差异

维度	Codex	DeepSeek Harness
开源	否（闭源）	是
模型自由度	锁定 GPT-5 Codex 系列	可自由替换任意模型
运行形态	云端为主（+ CLI）	本地 / 自托管 / CLI
定价	订阅制，成本较高	框架免费，仅模型 API 计费
可扩展性	官方工具为主	一切皆插件，可自写
数据归属	OpenAI 云端	用户可本地掌控

结论：Codex 是「最强闭源云端编程智能体」代表，能力成熟但贵、且不可控；Harness 是「开源、可换模型、可自托管」的替代路线。两者对标同一条赛道，但商业模式完全相反。

五维对比矩阵：一页看懂

5-6

维度	Harness	WorkBuddy	Trae Work	Qoder Work	Codex
开源	✅ 是	❌ 否	❌ 否	❌ 否	❌ 否
运行形态	CLI / SDK / 本地	桌面应用	IDE	编辑器 / 智能体	云端 + CLI
模型自由度	●●● 任意换	●● 官方若干	●● 内置为主	●● 内置为主	● 锁 GPT-5 Codex
插件/技能生态	●●● 一切皆插件	●● 官方功能	●● IDE 插件	●●● Skills 引擎	●● 官方工具
数据可控/本地	●●● 可自托管	● 云端	●● 本地 IDE	●●● 本地优先	● 云端
办公自动化	●● 需自建	●●● 强	●● 偏开发	●●● 强	● 偏编程
编程/开发	●●● 强	●● 中	●●● 强	●●● 强	●●● 强
上手门槛	中	低	低	低	低
价格	框架免费	订阅 / 企业	按 Token	订阅 / 企业	订阅（贵）
生态绑定	中立	腾讯系	字节系	阿里系	OpenAI

读表提示：竖着看选产品，横着看比能力。Harness 优势集中在「开源、模型自由、可扩展、可自托管」；劣势在「办公自动化、开箱即用」。

你的情况	推荐	理由
要完全可控、可二次开发、可换模型	Harness	唯一开源框架，模型可换，可自托管
重度使用腾讯办公生态、想省事	WorkBuddy	多专家 Agent，微信 / 腾讯文档打通
要一个全能 AI IDE 直接开干	Trae Work	AI 主导开发全流程，装好即用
数据敏感、要本地优先 + 阿里云生态	Qoder Work	本地优先，记忆 / 技能引擎成熟
预算充足、要最成熟的云端编程 Agent	Codex	闭源生态成熟，长时自主任务强
已有自研工具链、想嵌入 Agent 能力	Harness	不绑定编辑器，插件化嵌入



提醒：团队没有开发能力时，直接上 Harness 会踩「配置多、要写插件」的坑，此时更适合选 WorkBuddy 或 Qoder Work 这类成品。

1. **插件化 Profile 架构**：能力以「插件组合包 + patch 层」按需叠加，一套内核适配不同场景
2. **多入口形态**：同一配置支持 Web、Headless（无界面）、CLI 等多种运行方式
3. **深度可定制**：分层配置叠加（内置 → 用户 → 命令行覆盖），不改源码即可扩展
4. **原生多 Agent 编排**：内置子代理、工作流、长时目标、后台任务等协作机制
5. **可演进工程底座**：客户端插件热更新 + 独立 Web 壳，方便持续集成新能力

一句话总结优势：竞品卖的是「开箱即用的 Agent 产品」，Harness 给的是「可拆装、可换模型、可自建的开源 Agent 引擎」。

DeepSeek Harness 当前的短板

1. 仍是早期版本：目前版本为 0.1.0-rc.6（候选版），尚未到 1.0，接口与行为可能仍有变动
2. 定制门槛高：深度扩展需要理解 profile / 插件组合包 / patch 分层等概念，对非开发者不友好
3. npm 包只含编译产物：发布包里只有 lib/*.js 和 config，不含源码；要改 DSH 本身需回到完整仓库（GitHub）
4. 热更新有条件：客户端插件 HMR 只有在同时跑 pnpm run dev:web 时才生效，其余改动要重新构建并刷新页面
5. Web 壳不自带启动：apps/web 的 Vite 入口依赖 dsh web 注入 window.__DSH_BOOT__，不能当独立应用直接跑

适用边界：Harness 适合有开发能力、要做定制化 Agent、重视可控与成本的团队；不适合零技术用户、要开箱即用办公助手、依赖特定云厂商生态的场景。

科技媒体“技术领导力”简介：

科技媒体“技术领导力”矩阵，作为IT行业头部媒体，拥有80万科技领导者受众。成立于2019年，创作10W+爆文80多篇，100W+播放量视频30个，累计传播量破亿。

在科技领域具有广泛影响力，作为刘润年度演讲、吴晓波年终秀、陈春花新书发布会、江南春新书发布会特邀支持媒体。



账号矩阵：



覆盖平台：

微信公众号、视频号、抖音、小红书、知乎、B站、百家号、网易号



微信公众号：技术领导力



公众号：AI新猿人