How I Use Obsidian + Claude Code to Run My Life

Greg Isenberg

2026-05-24

目录

1 受访者简介 1

2 访谈主线 2

2.1 主题 1：为什么 Claude Code 单用「不够 game-changing」——context 才是真正的瓶颈 2

2.2 主题 2：Obsidian 到底特殊在哪里——不是 markdown，是「反向链接图」 3

2.3 主题 3：Vin 自建的 11 个 slash commands——把「reflection」变成可调用的功能 3

2.4 主题 4：`/trace` 实战——一个 idea 在 13 个月里如何演化 6

2.5 主题 5：`/ideas` 实战——从 reflection 到可执行的 build 6

2.6 主题 6：隐私与边界——Vin 的「单向写入」原则 7

2.7 主题 7：MD 文件是新的「氧气」 8

3 关键引语 8

4 方法论提炼：可被复用的 7 条规则 9

5 Vin 的立场地图 9

6 历史背景注释 10

7 延伸阅读 / 行动清单 11

1. 受访者简介

本期访谈对象 Vin（Internet Vin），独立创作者、播客主理人、长居多伦多。他在 X 与 YouTube 上以「计算机、笔记系统、agent 工作流」为核心做长期内容，2024 年底之前已有一套完整的笔记系统（Mac Whisper 语音转写 + Kortex 空间映射 + 实体笔记本 + Are.na 碎片库），2025 年 1 月才首次把 Obsidian 接入工作流。本次受访的直接背景是 Obsidian 官

方近期发布了 **Obsidian CLI**——这让 Claude Code 能够读取整个 vault 并理解 markdown 文件之间的反向链接关系，从而把多年的个人笔记变成 agent 可调用的「第二大脑」。Vin 在过去 13 个月里完整经历了从「怀疑 Obsidian」到「把它当作 agent 主源」的全过程，这次是他第一次公开演示自己实际使用的 commands 与 vault 结构。

主持人 Greg Isenberg 是 Late Checkout 创始人、Idea Browser 主理人，本次访谈他扮演「完全没用过 Obsidian 的中级 Claude Code 用户」，让 Vin 从零讲起。

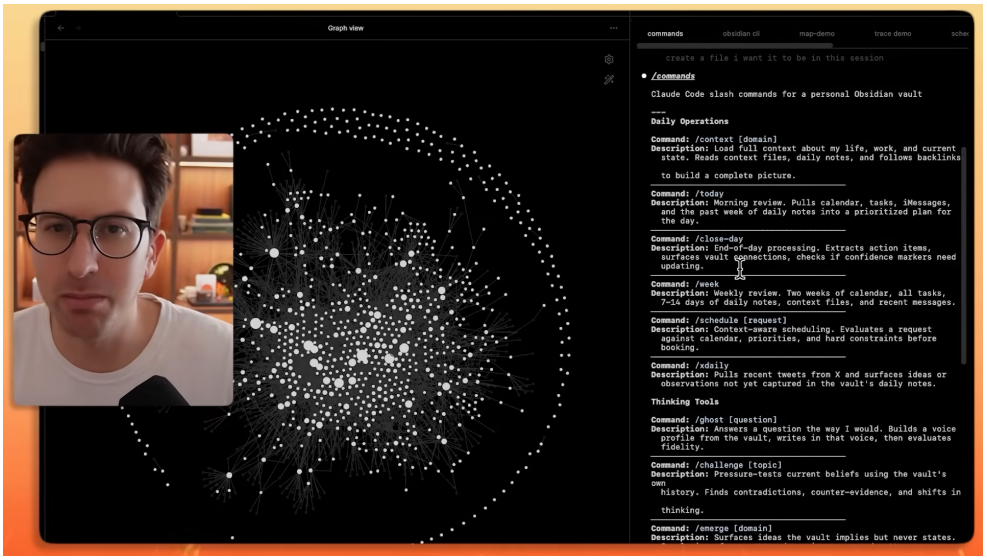

图 1: Vin 在多伦多家中演示 Obsidian + Claude Code 工作流

2. 访谈主线

2.1 主题 1：为什么 Claude Code 单用「不够 game-changing」——context 才是真正的瓶颈

问：Greg 想搞清楚一件事——为什么很多人装了 Claude Code，玩两天就觉得「也就那样」？Vin 的诊断是什么？

答：Vin 把 Claude Code 定义为「一个能通过自然语言控制你电脑的 agent」，然后立刻强调真正的瓶颈不在模型能力，而在**你给它的上下文**。他现场演示：让 Claude Code 在桌面建一个写着”hello Greg” 的文件——这一步很无聊，但他借此指出，agent 能力随着 context 的累积而**指数级增长**：「whatever you can describe to it, it can start to do」。

第二层，他没明说但反复暗示的判断是：网页版 Claude / ChatGPT 的 memory 是**不可控的黑盒**——你不知道它记住了什么、忘了什么。所以严肃用户必须把「记忆」外置成自己能看到、能编辑的文件。这是他后面整套方法论的前提。

第三层，他回避的部分：当 Greg 暗示「那你是不是觉得 ChatGPT memory 是个失败设计」时，Vin 没有正面攻击 OpenAI，而是用「you don’t know what’s in that memory」这

种中性措辞带过——他显然不想把这变成产品互撕，而想把听众注意力锁定在「文件 = 可控上下文」这个方法论上。

> *"The whole game is feeding the beast good context." ——Greg（约 04:30）*
> 上下文：*Greg 在 Vin 刚解释完「为什么要把项目描述存成文件」后接的一句，Vin 立刻点头确认。这句是整集的 thesis。*

2.2 主题 2：Obsidian 到底特殊在哪里——不是 markdown，是「反向链接图」

问：Greg 直球追问——Obsidian 看着就是个 markdown 编辑器，凭什么值得专门讲一集？

答：Vin 的回答分三层。

第一层是表层：vault 是一堆 markdown 文件，每个文件可以用 `[[ ]]` 链接到另一个文件。他当场演示在 daily note 里写「today I am on a podcast with Greg Isenberg」，Greg Isenberg 这几个字立刻变成一条到 Greg 专属笔记的链接。

第二层是为什么这个 trivial 的功能重要：他打开 graph view，展示**「Greg Isenberg」这个节点在过去几个月里被多少篇笔记反向链接到**——「notes on time constraints」「how I use Obsidian」等等。这意味着 vault 不是一个文件夹，而是**一张随你写作自动生长的知识图**。「A folder on your computer cannot show these interrelationships.」

第三层是真正的杀手锏：**Obsidian CLI** 让 Claude Code 不只能读到每个文件的内容，还能读到**文件之间的连接关系**。所以 agent 能做的不再是「总结这篇笔记」，而是「在跨越一年的几百篇笔记里发现你自己都没意识到的反复出现的模式」。

> *"It can surface patterns about what you're thinking about that you are not seeing for yourself … Some idea that you might have been writing about for a year in this vault, it could be a latent idea and it can just immediately say, hey, did you know that you've been writing about this same pattern." （约 13:40）*

2.3 主题 3：Vin 自建的 11 个 slash commands——把「reflection」变成可调用的功能

问：Greg 追问——这套东西具体怎么用？你每天打开 vault 之后干什么？

答：Vin 拉出自己写的 commands 列表（这些都是他亲手让 Claude Code 生成的 `.claude/commands/` 下的 markdown 文件，不是 Obsidian 内置的）。他按用途分两组讲解：

第一组：日常操作类

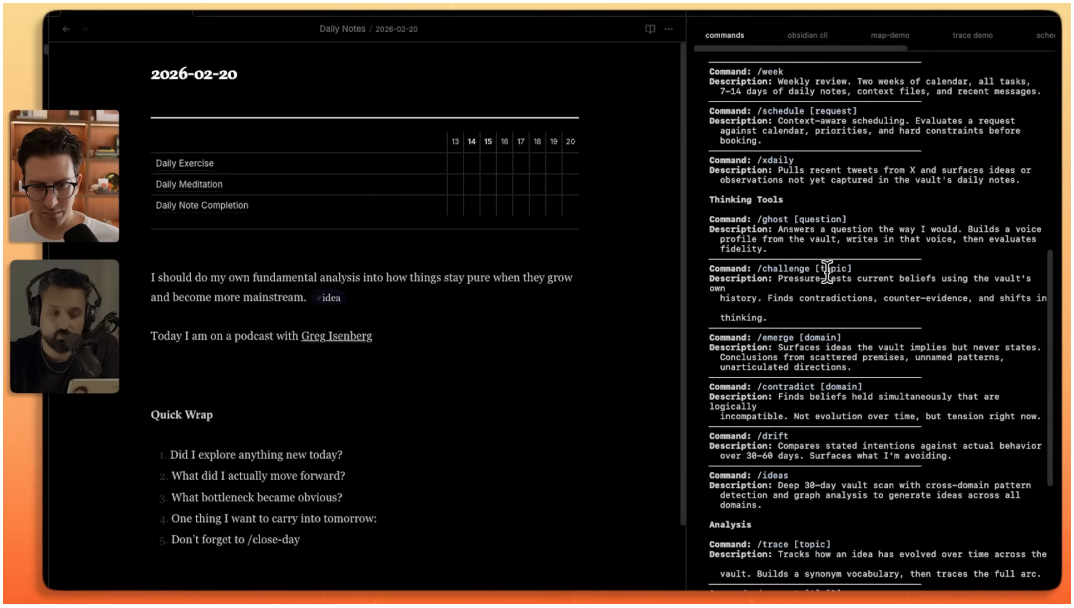

图 2: Obsidian graph view：一个概念节点的反向链接邻域，Vin 用它解释「vault 是图、不是文件夹」

命令	作用	为什么需要它
`/context`	读取 README、各项目 context 文件、最近 daily notes，一次性预加载	不用每次 session 重新解释自己是谁、在做什么
`/today`	拉日历、任务、过去一周 iMessage、daily notes，输出当天 prioritized plan	普通 agent 只看日历，看不到你最近在「想」什么
`/close-day`	提取 action items、surface vault connections、检查置信度标记	把一天的散落输入沉淀成结构化产出
`/schedule`	评估「能不能在 X 时间和 Y 见面」时同时参考日历 + 最近笔记里你在想什么	演示中它正确判断出「Greg 这集已经被你写了好几天，没必要再加一次会议」

第二组：思考工具类——Vin 反复强调「这才是我用 LLM 的主要方式」

命令	作用
`/ghost`	从 vault 建立你的 voice profile，用你的口吻回答某个问题，再评估 fidelity
`/challenge`	用 vault 自身的历史 pressure-test 你当前的观点，找矛盾、反例、立场漂移
`/emerge`	surface「vault 暗示但从未明说」的结论——把散落的前提汇成尚未命名的模式
`/drift`	对比过去 30–60 天「你说要做什么」vs「你实际在做什么」，揭示你在回避什么
`/ideas`	30 天 vault 扫描 + 跨域模式识别 + graph 分析，生成跨领域 idea（不只是创业，也包括该看的电影、该买的产品）
`/trace`	追踪某个 idea 在 vault 里如何随时间演化
`/connect`	给两个 domain，用 vault 的链接图找出它们之间的桥

他特别强调一个反直觉的事实：**这些 commands 大部分是 Claude Code 自己建议他写的**。流程是——Vin 先让 agent 评估「以我现在的水平和我在写的东西，你觉得我应该有什么 commands」，agent 提出后他再让它直接生成 command 文件。这是「让 agent 设计自己的脚手架」的一个很干净的范例。

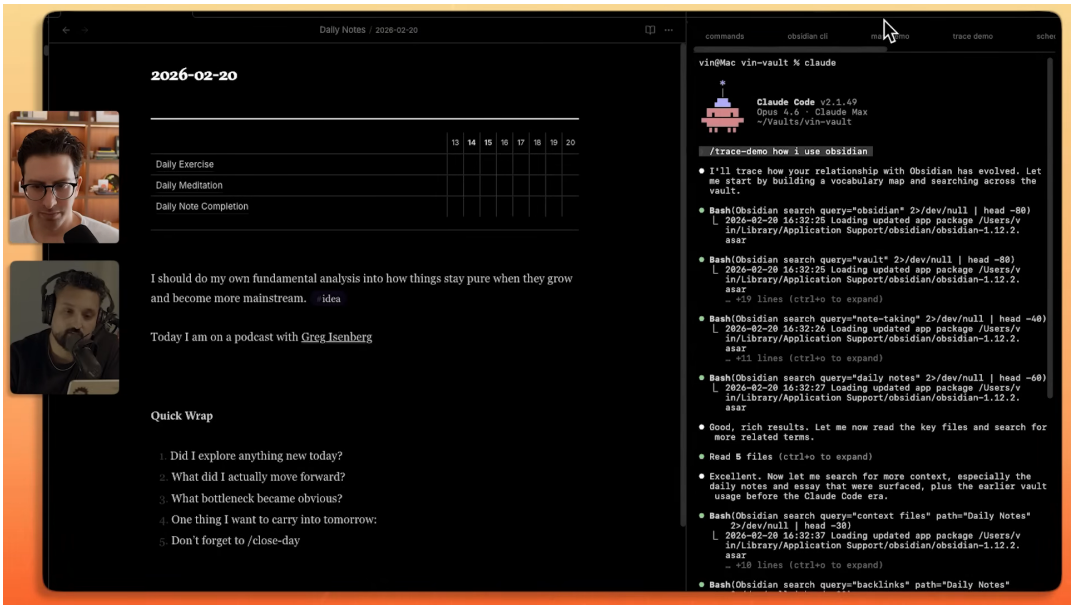

图 3: Vin 的 `.claude/commands/` 目录截图——11 个自定义 slash command 全是 markdown 文件

2.4 主题 4：`/trace` 实战——一个 idea 在 13 个月里如何演化

Vin 现场跑 `/trace` 追踪「我和 Obsidian 的关系」如何演化。输出是按时间分阶段的报告：

- ☐ **Pre-vault（2024.12）**：在文章「How I take notes in West End Toronto」里详述完整笔记系统，Obsidian 完全不在画面里。
- ☐ **Discovery & Skepticism（2025.1–5）**：第一篇 daily note 里写「bidirectional linking is not that useful, but I don't know」——明确表达怀疑。
- ☐ **方法论转折点**：在某篇笔记里他自己写下「最有用的不是给『播客』『健身』这类通用 tag 加链接，而是**为我每个 pattern / theory / project / perspective 建一个独立 note**，再链接到这些 note」。
- ☐ **Phase 4（2026.1）**：「a month of explosive building」——摩擦点已经不在 Obsidian 本身，而在「vault 和 agent execution 之间的边界」。

Vin 的点评：「I would not be able to do this on my own and this fast」——这不是模型炫技，而是一种**个人考古学**，让你看见自己作为一个时间序列的样子。

> *"I just think it's so cool that … we can work with these agents and we still have to explain things to them, but we only need to explain them once."*（*约 47:20*）

2.5 主题 5：`/ideas` 实战——从 reflection 到可执行的 build

Greg 提出关键质疑：**你举的例子全是个人反思，这跟「建东西」有什么关系？**

Vin 直接跑 `/ideas demo`。命令执行了约 5 分钟（他强调：用了 vault 之后所有请求都变慢，因为读的文件多到夸张）。输出是一份分类报告，关键段落：

Tools to build

- ☐ `/graduate` command：扫 daily notes，识别被 tag 或未被 tag 的 idea，提示用户决定「升级为独立 note / 并入已有文件 / 丢弃」——把 daily stream 转成结构化 idea pipeline
- ☐ 给团队建一个共享 Obsidian vault for 「Other Stuff」播客
- ☐ 强制 day-per-domain 的 time blocking app

Systems to implement

- ☐ Inline delegation：直接在 daily note 里写「schedule a call with X about Y this week」，让 Otis / Claude bot 自动拾取并执行

Conversations to have：报告甚至列出了具体应该找的人名（Aaron、Steph Ango——Obsidian CEO 等），并给出理由。

压轴一幕：Vin 直接对 agent 说「build the `/graduate` command」，agent 立刻生成了完整的 command 文件，slash 命令当场可用。这是访谈里最具说服力的片段——**从 reflection 到 idea 到自我扩展工具的闭环，时间不到一分钟**。

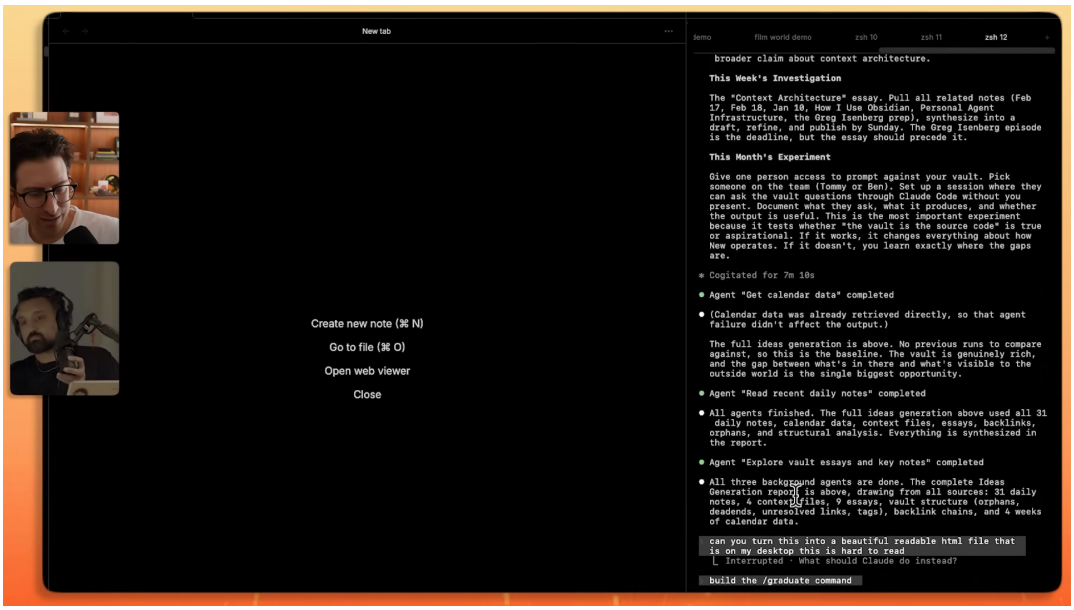

图 4: `/ideas` 输出的报告：tools to build / systems to implement / conversations to have 三栏

2.6 主题 6：隐私与边界——Vin 的「单向写入」原则

问：Greg 直接挑明——把 Obsidian 当第二大脑很好，但把第二大脑的钥匙交给 Claude Code 很恐怖。你怎么想？

答：Vin 承认这是「the fundamental weird element of this technology」，他没有粉饰。然后说出全集最重要的方法论之一：

Vin 的硬规则——agent 只读不写。他不让 Claude Code 往 vault 里写文件。理由是：「if it starts making its own files in this vault, then ... is it finding patterns about things it's written or is it finding patterns about things I've written?」——一旦 agent 的输出污染了 vault，未来所有 pattern recognition 就失去了「这是我的思想」这个前提。

agent 生成的内容他放在 vault 外的临时文件里看，自己手抄需要的部分进 vault。这是一条非常重要的工程纪律，但大部分人不会想到。

他也承认自己刻意给了 agent 大量个人信息，「purposely」——理由是他想理解这项技术究竟在改变什么，但他同时也警告听众：「you have to really think about how much information you're sharing.」

2.7 主题 7：MD 文件是新的「氧气」

访谈最后 10 分钟两人合力打磨出一句新口号。Greg 说：「people think tokens are the oxygen, but they're not, markdown files are the memories.」Vin 立刻接：「a file is essentially perfect memory」——人类记忆会美化、会扭曲（Greg 举他们一起去 Mississauga 剪头发那次为例），而 markdown 文件在被写下那一刻就被冻结，未来 agent 调用时拿到的是一个**无衰减、无美化**的数据点。

两人在这里达成的共识可以提炼为：**在 LLM 时代，一个人的「资产」不是他和 ChatGPT 聊过多少次，而是他有多少结构化、可被 agent 索引的 markdown 文件**。

3. 关键引语

> *"The quality of information that the agent has entirely determines what it can do for you."* ——*Vin*（约 *39:50*）
> 上下文：*Greg* 总结「*agent* 有多最新的信息决定它能做多少」时 *Vin* 的回应，态度坚定，是他整套方法论的第一性原理。

> *"I really, really, really like working with LLMs as a thinking partner. That's my favorite way of using LLMs."* ——*Vin*（约 *18:30*）
> 上下文：在介绍 *thinking commands* 之前的自我定位。他刻意把自己和「用 *agent* 建产品」的主流叙事拉开。

> *"I don't want it to make a file … I want to control all the files in my obsidian vault because I always want to pull from what I think about things, not what it thinks about things."* ——*Vin*（约 *36:15*）
> 上下文：回应 *Greg* 关于「*agent* 是否应该往 *vault* 写东西」的问题，态度非常坚定。这是他和「*agent everywhere*」派最明显的分歧。

> *"If I'm OpenAI or Anthropic, I'm buying Obsidian. It's the missing link."* ——*Greg*（约 *50:20*）
> 上下文：看完 `/ideas` 输出之后脱口而出。这是访谈最高潮的判断——也是 *Greg* 第一次明确把 *Obsidian* 抬到「战略级基础设施」的位置。

> *"There's no excuse anymore for me not to be writing down and reflecting into markdown files."* ——*Greg*（约 *51:00*）
> 上下文：访谈结束前 *Greg* 对自己说的话，反讽地宣告他过去一直在偷懒。这是访谈作为「劝退/劝入」内容最成功的一刻。

> *"We are potentially watching a fundamental shift in the human relationship to computers."* ——*Vin*（*约 49:30*）
> 上下文：在被问「这一切意味着什么」时给出的回答。语气罕见地变慢、变正式。

4. 方法论提炼：可被复用的 7 条规则

这部分是访谈里散落但可以抽出来的工程纪律，不是 Vin 列表式给的，而是他展示中体现的：

1. **把项目描述存成文件，不要存进 chat 历史**——session 是 ephemeral 的，文件是 persistent 的。
2. **为每个 pattern / theory / project / perspective 建独立 note**，而不是给「播客」「健身」这种通用 tag 建索引——后者是 Vin 早期走的弯路。
3. **建一份 `personal-workflow-context.md` 和每个项目独立的 `<project>-context.md`**——agent 通过 `/context` 一次性加载。
4. **让 agent 自己设计 commands**——你告诉它你的水平和目标，让它提议合适的脚手架。
5. **Agent 只读不写 vault**——保持「这是我的思想」这条前提不被污染。
6. **给每个 idea 打置信度标签**（Vin 用 confidence markers）——便于 `/close-day` 和 `/challenge` 后续追踪立场漂移。
7. **接受请求变慢**——读几百个文件就是会花 5 分钟，这不是 bug，是 context 深度的代价。

5. Vin 的立场地图

把 Vin 在这次访谈中的立场，与他过往（X / YouTube / 他自己 vault 引用的旧文章）公开发言对照：

议题	本次访谈立场	过往公开立场	是否转变
Obsidian 的反向链接是否有用	极其有用，是整套 agent 工作流的基础	2025.1 daily note：”bidirectional linking is not that useful, but I don’t know”	是（13 个月内 180° 转变）
用 LLM 的主要方式	thinking partner（反思、模式识别）	一致——他多年公开内容都强调写作和反思	否
Agent 应否写入个人 vault	强烈反对，单向只读	此前未公开讨论过，本集首次明确	新立场
把多少个人信息交给 agent	「purposely」交了很多，但承认这是 weird element	与他长期「极限实验」人设一致	否
Note 系统的核心抽象	不是文件夹，是**图**（节点 + 反向链接）	早期他用 Are.na + Kortex 做空间映射，已经在追求图状结构	微调（工具变了，philosophy 一致）

最值得注意的转变是**对反向链接的态度**——13 个月前他在自己 vault 里明确写「我不确定这有用」，13 个月后他正在用它跑全集最炫的演示。这个转变之所以重要，是因为它说明**方法论的胜负不取决于第一直觉，而取决于愿不愿意持续记录到能让自己被自己说服**。

6. 历史背景注释

- **Obsidian**：2020 年发布的本地 markdown 笔记工具，免费，开源插件生态，核心特性是 `[[wikilink]]` 反向链接和 graph view。CEO 是 Steph Ango（Vin 在 `/ideas` 输出里把他列为应该聊的人）。
- **Obsidian CLI**：本集的技术前提。这是 Obsidian 团队近期发布的命令行接口，让**外部程序（包括 Claude Code）**能读取 vault 内容**和反向链接关系图**。没有这个东西，本集的所有演示都不成立。

- **Claude Code**：Anthropic 官方 CLI，2025 年初推出。和 Cursor / Aider 的区别是它默认在终端跑、能直接读写本地文件、支持 `.claude/commands/*.md` 形式的自定义 slash command。
- **Mac Whisper / Kortex / Are.na**：Vin 在 2024.12「How I take notes in West End Toronto」中提到的旧栈——分别用于语音转写、空间映射、碎片收集。理解这些工具的存在，能解释为什么他对 Obsidian 起初冷淡（他已经有一套不依赖反向链接的成熟系统）。
- **Granola**：AI 会议笔记工具，Greg 在追问「能不能把会议记录灌进 vault」时举的例子。
- **Toby (Shopify 创始人) / Aaron Stadium / Black Mountain College**：Vin vault 里反复出现的人名/概念，本集 `/ideas` 和 `/connect` 输出里也提到。理解他在多伦多搞 Stadium 这个物理空间的背景，能解释他对「physical space + 文档化」这条线为什么这么执着。
- **Idea Browser**：Greg 自己的产品，每天推送一个 validated startup idea。Greg 在访谈中点出一个用法——把每日推送灌进 vault，让 Claude Code 基于此建东西。

7. 延伸阅读 / 行动清单

- ☐ **Vin 的 skills 和 Greg 的笔记**：访谈描述里给的链接 `startup-ideas-pod.link/obsidian...` ——直接拿到 Vin 实际用的 commands 模板
- ☐ **Obsidian 官方文档 + Obsidian CLI release notes**：理解 CLI 暴露的 API 才能写自己的高阶 commands
- ☐ **Vin 旧文：How I take notes in West End Toronto (2024.12)**：理解他「pre-Obsidian」系统，对比看出他到底因为什么转向
- ☐ **Steph Ango 的博客**：Obsidian CEO，长期写「file over app」「software as a place」，是 Vin philosophical 取向的源头之一
- ☐ **自己的最小行动**：今天就建一个 vault，先只用 daily note + `/context` 一个 command，跑两周再加复杂功能——Vin 反复强调他自己花了 13 个月才到现在这一步

图 5: 访谈尾声：Greg 宣布「没有借口不开始写 markdown 了」