

VS Code 擴充套件 開發實務報告： VirtualTabs 與 AI 協作經驗

從個人痛點到 1,400+ 下載量的
開源專案實證

本報告旨在彙報 VirtualTabs 的產品解決方案、技術突破、
營運策略以及 AI 輔助開發的量化成效。



專案名稱

VirtualTabs
(VS Code Extension)



核心價值

解決大型專案 Context
管理，實現「邏輯分組
優於實體結構」。



市場驗證

累積 1,400+ 下載量
(VS Code Marketplace /
Open VSX)



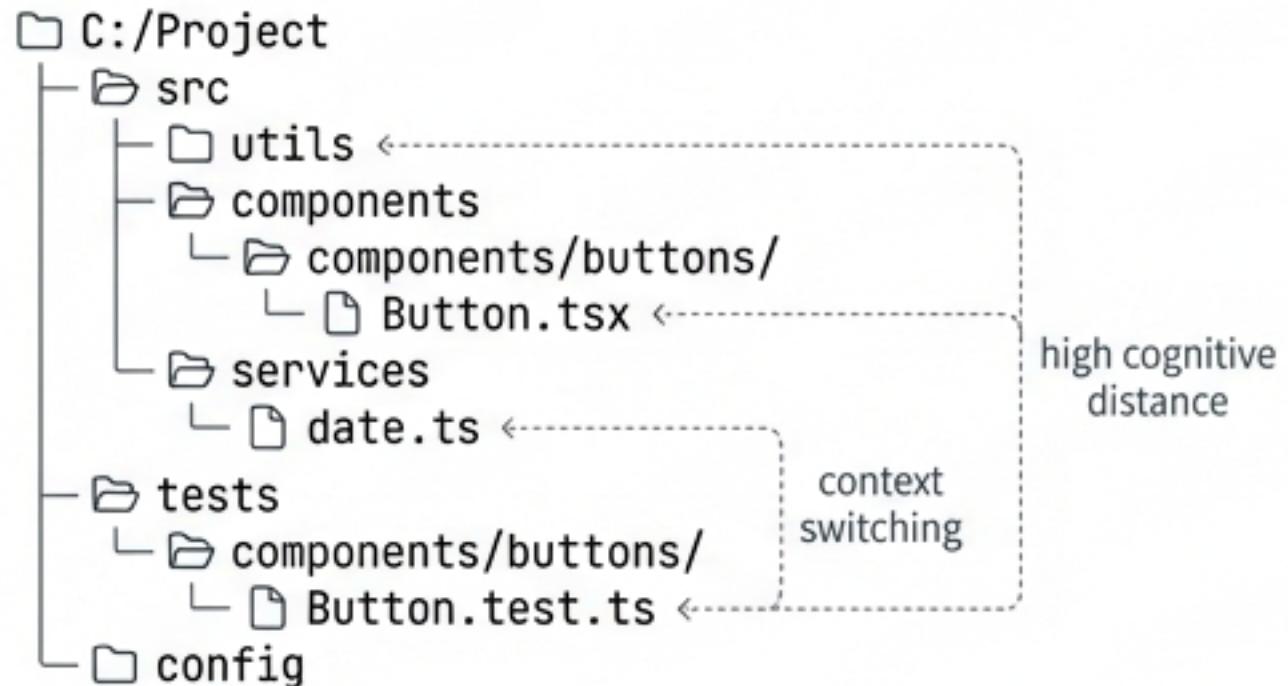
開發模式

AI Native Workflow
(單人開發，團隊級交付
速度)

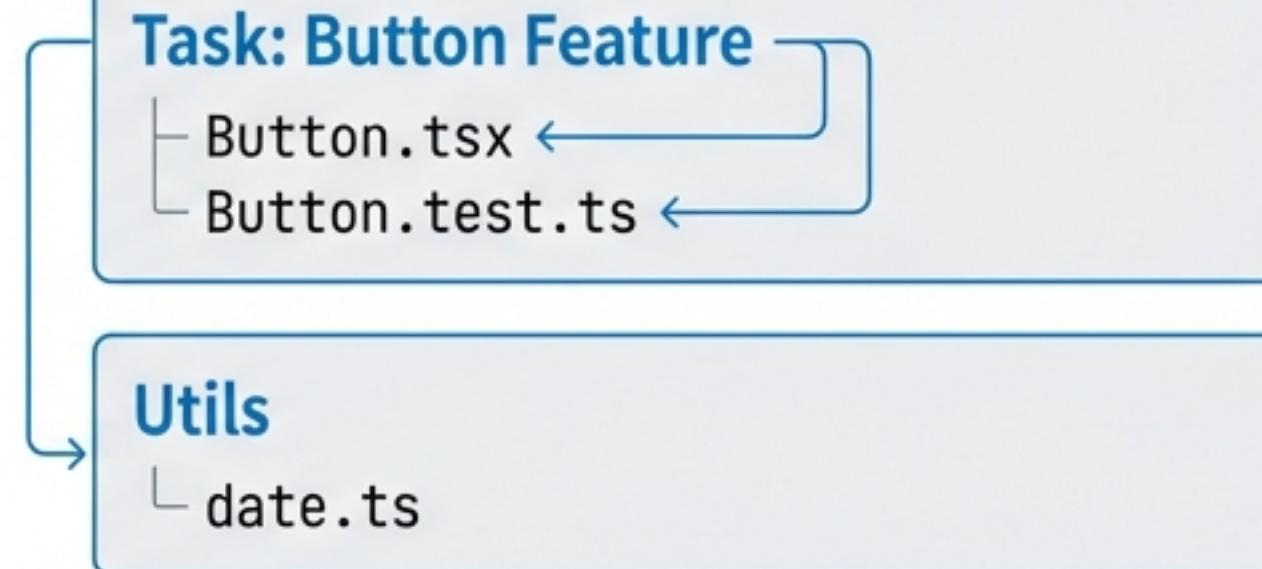
產品解決方案：解耦實體目錄，重塑開發認知負荷

痛點：在 Monorepo 或中大型專案中，相關檔案（Controller, Service, DTO）散落在不同深層目錄，導致高昂的認知切換成本。

物理檔案系統 (Physical) – AS IS



VirtualTabs 邏輯分組 (Logical) – TO BE



Workspace Management

跨目錄組織檔案，支援「巢狀群組 (Sub-groups)」以應對複雜業務邏輯。

Context Engineering

獨創「Copy Group Context」功能，一鍵將邏輯群組轉換為 LLM 友善的 Markdown 格式。

Dev Tools

整合 Inline Actions (直接執行 .bat/.sh 腳本) 與書籤系統，優化「編輯-測試」循環。

技術實作突破：AI 協作下的演算法與 API 整合

利用 AI Agent 突破 VS Code API 限制並加速核心邏輯交付。

Technical Case 1: 介面限制突破 (Drag & Drop)

- 挑戰：**VS Code TreeView API 原生不支援外部資料夾拖曳夾後的自動遞迴展開，且 MIME 解析繁瑣。
- AI 解決方案：**透過 AI Agent (Antigravity) 生成 DataTransfer 擋截邏輯與路徑解析骨架。
- 成效：**將原本需耗時數天的 API 文件查閱縮短至數小時完成。

```
1 // AI Assisted Logic
2 const uriList = dataTransfer.get('text/uri-list');
3 if (uriList) {
4     const uris = uriList.value.split(/\r?\n/);
5     // Recursively expand paths...
6 }
```

Technical Case 2: 資料完整性 (Circular Dependency)

- 挑戰：**實作巢狀群組時，必須防止「A 包 B 且 B 包 A」的無限迴圈導致崩潰。
- 演算法實作：**採用 DFS (深度優先搜尋) 演算法進行 `isDescendant()` 檢查。
- 開發模式：**AI 產出完整演算法核心，人工僅需專注於邊界防錯與持久化驗證。

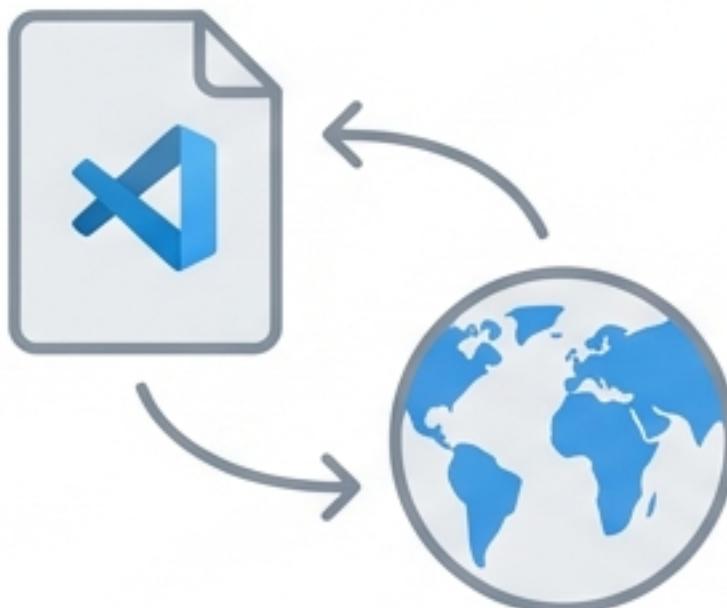


```
private isDescendant(groupId, potentialAncestorId): boolean
```

營運與生態系：結構化開源與人機雙向 SEO

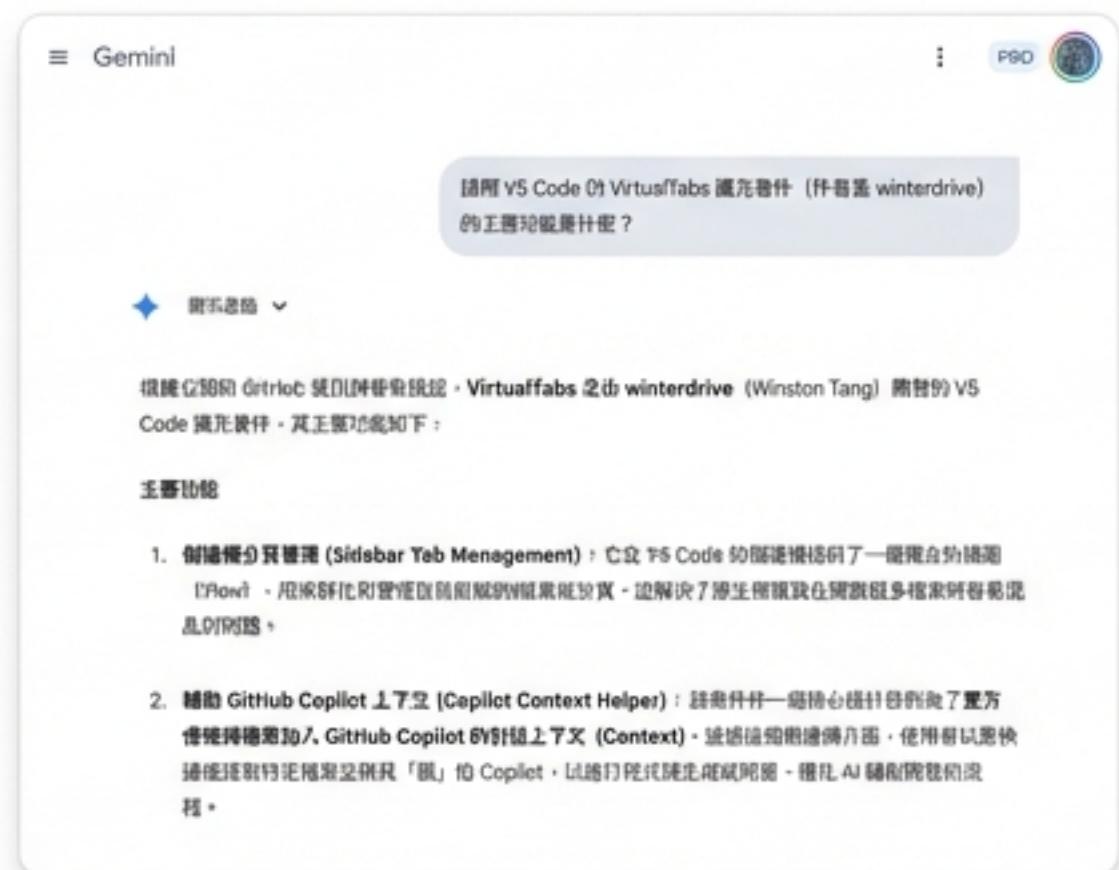
Strategy 1: 基礎建設即配置

- **多語系支援 (i18n)**：同步維護英、繁、簡三語系文件。
- **團隊共享**：透過 `.vscode/virtualTab.json` 實現配置 Git 管控。



Strategy 2: AI SEO (人機雙向發現機制)

- **創新部署**：率先部署 `llms.txt` 規格，建立 AI Agent 專用的純淨文檔入口。
- **實測成效**：Gemini 等模型已能精準識別專案作者與核心功能。



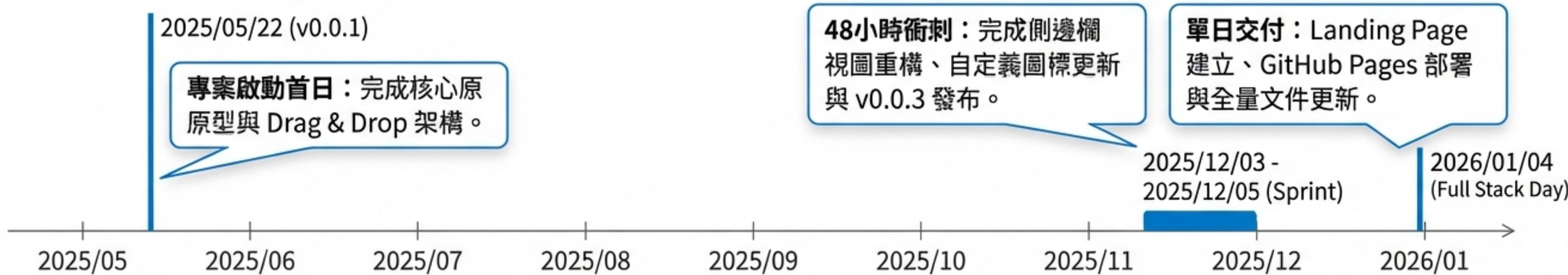
Strategy 3: 全球社群驗證

- R3flector (俄羅斯)：推動 Sub-groups 核心重構。
- E-michu (波蘭) & swhitlow-gmtv (美國)：協助優化排序與穩定性。
- jianfulin (台灣)：貢獻 Delete Confirmation 功能與導師級 Code Review。



交付數據與成效：單人開發，團隊級速度

High-Density Delivery Timeline



Metric Verification

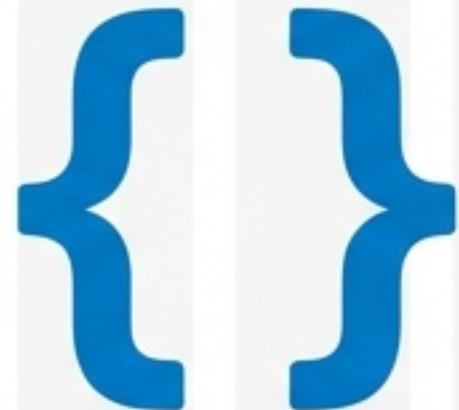
側專案驗證 (QuickPrompt)

驗證 Prompt 模板化需求，並在 Git Log 中記錄到 AI Agent (copilot-swe-agent) 直接貢獻程式碼的實證 (2025-12-05)。

Summary

透過 AI 處理 Routine Work (翻譯、文件、樣板代碼)，原本需 3-5 天的整合工作縮短至「單日交付」。

總結與展望：開發者角色的典範轉移



Writer (代碼編寫者)

AI Leverage



Verifier (方案驗證者)

Key Takeaway: 價值錨點的改變

開發者角色正從實作轉向架構驗證。AI 將實作成本降至最低，使開發者能將精力集中在 UX 細節、架構彈性與解決方案驗證。

Future Roadmap

- 持續優化 AI Context 策展功能。
- 深化 llms.txt 應用，探索更精準的 Agent 協作模式。
- VirtualTabs 案例證明：個人開發者透過 AI 協作，能夠維持完整的 DevOps 流程。