

HWDC 2026 • 2026-10-15 16:00 • F 會議室

AI 把寫程式變便宜了， 那開發者還剩下什麼？

鄭重男 Nickle Cheng

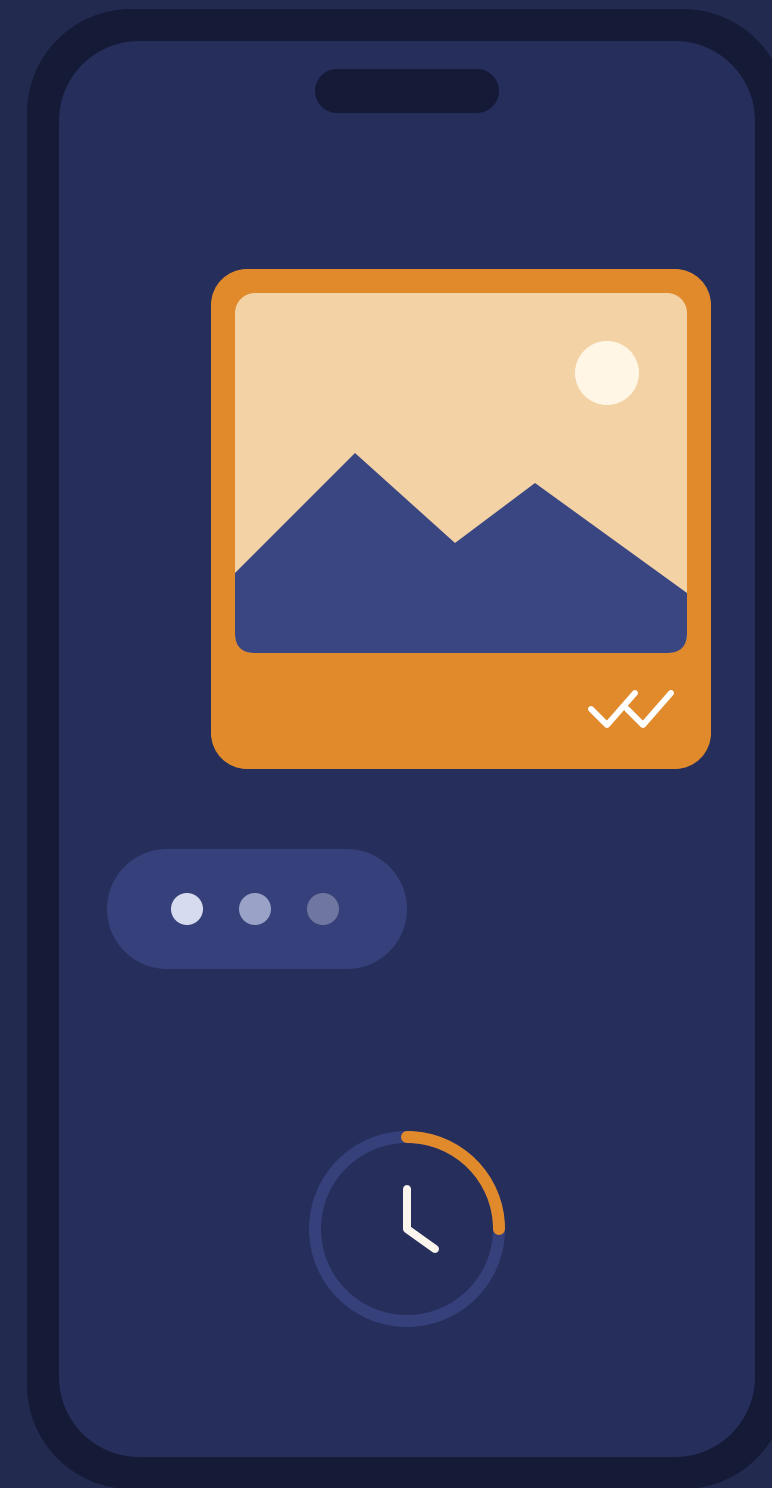
國軍高雄總醫院 資訊室主任

Vibe Coding

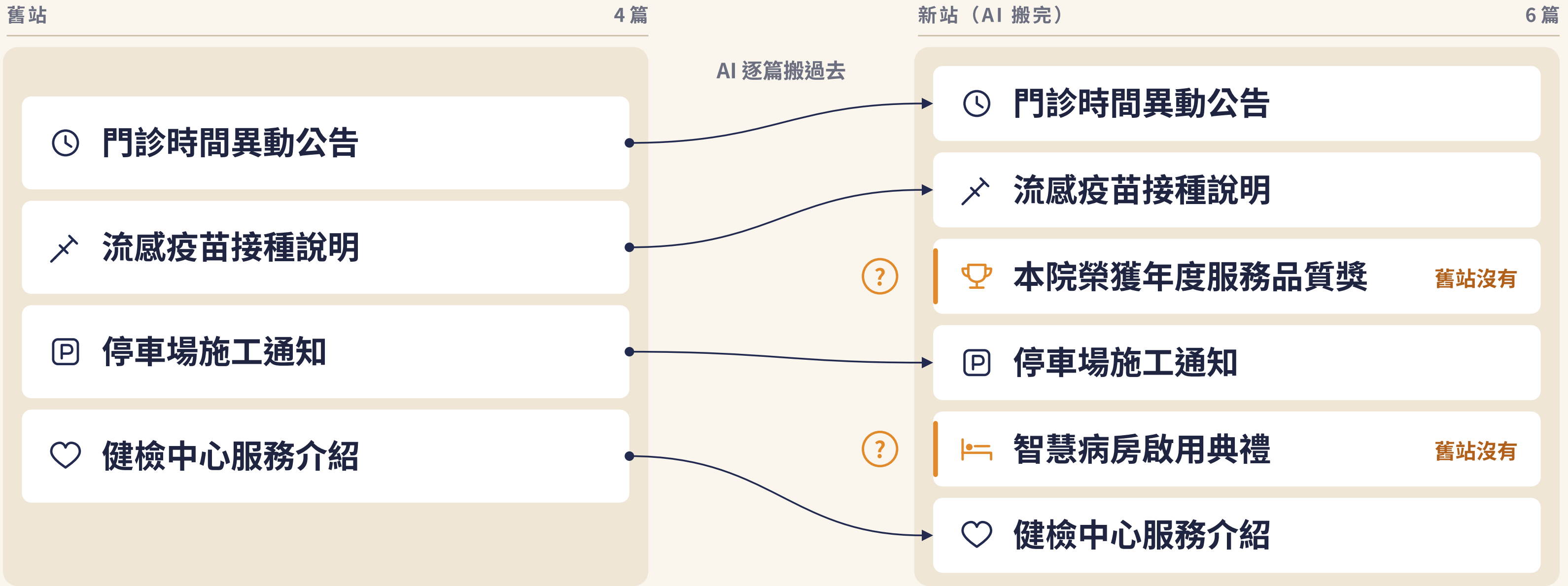


來拍張合照

等一下回來看



我的官網，長出了一堆不存在的文章



誰核准新增？AI 產出得越快，人類越難逐一驗證

排版排得越漂亮， 越容易忘記問 這是真的嗎？



好看的版面

✓ 能跑 ✓ 能讀 ✓ 很順 ≠ ② 對的 ② 能上的

■ 記住一個細節：我抓得到，是因為我知道舊站上本來有什麼

便宜的東西不會是瓶頸， 瓶頸只會往還很貴的地方移動



問題不是「AI 會不會取代開發者」
當寫程式不再是門檻，什麼才是真正的價值？

示意比例，非實測數據

四個重點

我們常聽到的答案，都落在中間兩層



Agenda

要講四件事



真的在跑的東西把它們串起來

一 • 想法

AI 會給你「怎麼做」 但不會給你「該做什麼」

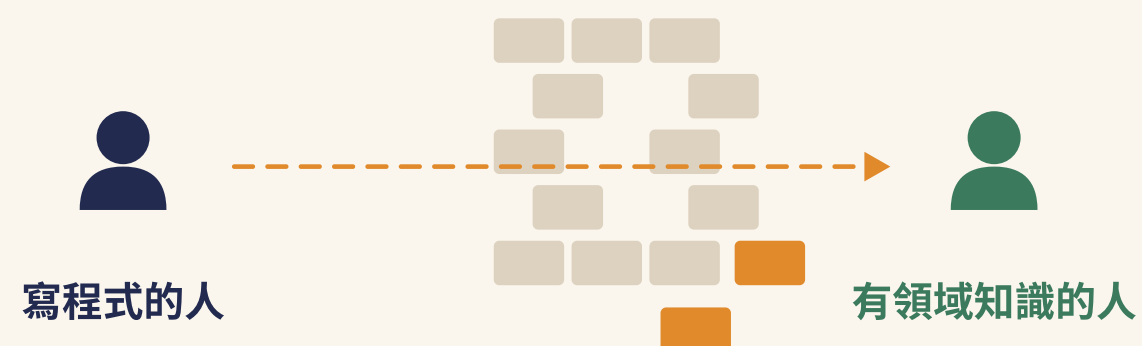
你問AI怎麼做

AI 能給你很多方法
但不代表每個都適合



該做什麼

答案不只在程式碼裡
是在真實場域



懂技術的人不一定懂現場
懂現場的人不一定會寫程式

AI 正在打破這道牆

前後端分離，專業分工 AI 更適合全端工程師？



決定「不做什麼」



AI 可以幫你做更多，但不能替你決定哪些不該做
你要求更多功能，它就持續擴充



在醫院，我最常做的決定反而是
「這個先不要做」



每增加一個功能
就多一份維運成本與法規責任

功能清單

- ✓
- ✗
- ✗
- ✓
- ✗
- ✗

決定你能做出什麼的，是你對現場的理解
而不只是你會什麼語言

二 • T Shaped • 先看五秒

【戰場設定：H-Hour Baseline】

演練時間點：116/3/12 上午 09:00

假想敵（Threat）：新興呼吸道傳染病

≡

傳播途徑：飛沫（空氣）及接觸傳播

🕒

潛伏期：2 到 7 天

📋

臨床症狀：發燒、咳嗽、呼吸困難

【當前戰況（Current Status）】

社區傳播已持續擴大，本市已達 23 例本土確診，並伴隨 3 起校園與長照機構群聚感染事件

研判：未來 72 小時內病例將快速暴增（Surge Warning），請各權責單位於今日中午前完成

DEMO / 虛構演練情境 本圖為示意，不對應任何真實單位或事件

你有看出問題嗎？

T 型

11 / 30

看起來完成，不代表真的完成

演練投影片



兩個問題

- 1 右邊三組圖表，到底在比較什麼？**
沒有指標、沒有單位、沒有數據來源
卻畫得像真的分析圖表
- 2 左下角那兩行字，右邊那段哪去了？**
被病毒圖蓋住的，正是
「請各權責單位於今日中午前完成回報」

**從 AI 生成到正式交付
少了最重要的一步：人工審查**

看起來越完整，越容易忽略其中的問題

不要把自己無法審查的東西交給 AI

深度的用途變了



以前
懂得深，所以寫得出來

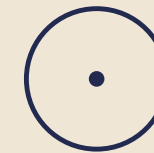


現在
懂得深，所以知道它哪裡在唬爛

AI 是乘法，不是加法



圓的大小 = 審查半徑

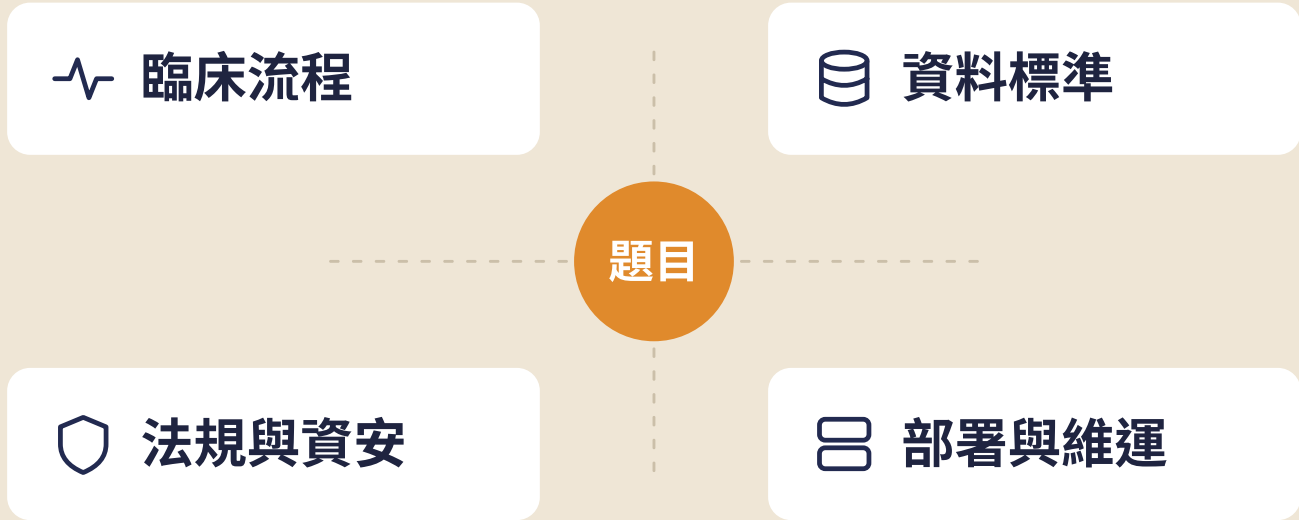


同一個 AI，在不同人手上結果差很多
差的不是提示詞，是那個人的**審查半徑**

回到官網：我抓得到，只因為我知道舊站上本來有什麼

不只把自己做好，還要把別人串起來

我這份工作要接的東西



沒有一塊是我最深的，但少任何一塊就接不起來。
題目常常長在交界上。

OMARCHY • SPECTRA 外掛

2.5 小時

在不熟的 Omarchy 桌面，從複製內建時鐘範本開始，做出狀態列的 Spectra 專案面板，並上架到官方外掛市集。

複製範本 → Spectra 面板 → 上架市集

前端加桌面殼層，兩個都不是我的一豎

深度自己練，廣度 AI 加速

一豎

AI 代替不了你的判斷

專業深度讓你能定義問題、判斷風險
也能審查 AI 的成果

自己練

一橫

AI 幫很大

跨進新領域的門檻
大幅降低了

AI 加速

AI 幫你走得更廣 專業讓你看得更深

T Shaped

T 型人才的審查能力



廣度 × 深度 = 審查面積

不只要懂得深，也要看得廣

AI 幫你跨越越多領域，你就越需要
跨域審查的能力

你能審查多少，取決於你的專業深度
與跨域廣度

不會寫程式但 Vibe Coding 出軟體 可以應徵軟體的工作嗎？

放到哪一格，不是看你能不能做出來，是看你審查得了多少？



● ● ● —————> ● 錯誤代價越高，越需要有能力審查與負責

結果換算成可以放手的刻度

決定位置的三個判準

- 1 出錯的代價多大
- 2 能不能回復
- 3 會不會影響重要決策



● 影響越大、風險越高，就越不能完全放手

哪裡該放手？

	階段	我實際放手到哪	為什麼是這個程度
1	≡ 需求分析	直接跟 AI 討論，一輪一輪討論到收斂	含糊的需求在討論裡就會現形；砍不砍，最後我拍板
2	品 架構設計	AI 提方案，我決定取捨	每個方案都有道理，但選擇需要判斷
3	</> 程式開發	大幅放手，遵循既有慣例	有慣例可循，偏掉了看得出來
4	✓ 自動化測試	放手讓它寫，驗收條件我自己定	測試全部通過，不代表需求真的被滿足
5	🔒 文件維護	文件可以給 AI 寫，知識交接不能放手	這裡決定下一個人能不能接手

我做的四件事

01



原型用合成資料

先驗證想法，
不必一開始就接觸真實病歷

02



上線責任不外包

誰核准上線、誰承擔責任，
必須事先明確

03



驗證強度綁在後果上

不看程式漂不漂亮，而看錯誤會造成什麼影響

04



可稽核的痕跡

決策、核准與測試結果，
要有正式紀錄

不只是自動化，而是重新設計工作流程



以會議逐字稿為例

痛的不是與會者是承辦人

 第一代

人工整理
錄音、反覆重聽、自己寫摘要

錄音 — 重聽 × N — 寫摘要

60 分鐘的會，
至少吃掉 2 小時

 第二代

雲端轉寫，
匯出、上傳、等待、人工整理

拿檔 — 上傳 — 等 — 整理

拿檔、上傳、等、整理，
手動那幾步全部還在

 第三代

流程整合
直接收音、本地端辨識與整理

散會 — 直接產出

散會後，直接取得會議紀錄初稿

 每場 60 分鐘的會議，可望省下約 **120 分鐘**

第二步 · 串流程

模型會做，不代表系統能用



1 音源要共用，不能互相干擾

換一個會議室就要現場實測，
看規格書不算數

2 散會後，自動完成後續處理

手動上傳那一步一存在，
時間成本就偷偷跑回來

3 本地端是限制，不是偏好

會議內容不能離開院內，
所以從一開始就決定了整套架構

第三步 · 做產品

哪些可以共用，哪些需要因地制宜？

四個固定 · 資料與推論不離開院內



收音操作一致



散會自動觸發



資料與推論留在院內



摘要優先於逐字稿

已經有其他醫院，
準備用這套流程進行 PoC

從單一場域的解法，走向可複製的應用

五處接縫 · 搬過去要先問什麼



會議室硬體

現有音訊鏈長怎樣？音源會不會被獨佔？



推論算力放哪

本機，還是院內推論服務？那條線算不算離開院內？



承辦人流程

摘要交給誰？要不要簽核？存在哪？



輸出格式

決議、待辦要不要拆開？有沒有固定公文格式？



語言混雜程度

中英、中台夾雜跟台語，辨識率明顯較低。

這還沒解決

不放手的，是決策背後的脈絡

</> 程式碼

👁 AI 讀得到

沒被寫進程式碼的那層脈絡

🚫 AI 讀不到

- 🕒 三年前為什麼選了這個怪怪的做法
- ⚡ 哪個單位踩過哪個雷，不能再犯
- 📄 這個欄位的規則，其實是某次跟法遵吵完的結果

我原本的想法

~~只有我知道得越多，就越不容易被取代~~
(其實只是讓別人更難接手)

真正該寫下來的，
是當初為什麼這樣做

讓下一個人接手，不必再踩一次雷

沒留下紀錄的決策脈絡，AI 也無從得知

回到開場那張照片

還記得剛才那張照片嗎？



關卡 01 · 你丟

一則訊息就啟動

- 手機丟一則截圖、連結或文字
- 不用開電腦，打一句就走

↑ 網址回到頻道

關卡 04 · 建立草稿

送上你的部落格

- 先檢查有沒有寫過
- 整理成文章格式
- 存進網站資料庫
- 新文章預設不公開

這兩關全自動，人不用在場

關卡 02 · 接手

常駐小主機接手

- 訊息一到就開工，不用叫醒
- 開一個隔離的工作間
- 只看得到你允許的資料
- 密碼不交到 AI 手上

↓

關卡 03 · 整理

讀完寫成文章

- 連結整篇讀完，截圖直接看懂
- 拆成一則一則的主題
- 自動分類、貼上標籤
- 跟舊文章互相連起來

我只傳了一張照片，剩下的，AI Agent 自己完成

從 Writer 到 Maker

“*It's pencils down on hand-written code, the best programming language in the world is now English, and I've retired as a software writer. The future belongs to software makers.*”

— DHH LinkedIn • 2026-09-29

親手寫程式，該放下筆了。全世界最好的程式語言是？

不再當 ~~寫軟體的人~~，而是做 打造軟體的人

從寫軟體的人，到做軟體的人

從 Writer 到 Maker，放手仍要看後果

✓ 同意

寫程式變便宜是真的
瓶頸移動也是真的

✗ 不同意

把「便宜」直接推成「一律放手」，中間少了一步：

成本降低



評估後果



分級放手

保留人審查。差別不在審不審，在審查強度要不要依後果分級

做軟體的人，還剩下什麼？



01

想法

題目從現場來



02

T 型

你審查得動



03

光譜

依後果決定放手



04

交得出去

留下決策，讓別人接得手

回去對你手上任何一個專案問

五個問題

01 這個題目是從哪裡來的？
有哪個功能我今天就該砍掉？

想法

02 這段東西我審查得動嗎？
審查不動就不要放手

T 型 · 一豎

03 不熟悉的環節，我接得起來嗎？
缺的那塊，現在補起來快很多

T 型 · 一橫

04 這段錯了會怎樣、誰要扛？
照答案決定驗證多嚴，長大了把關跟著長

光譜

05 如果我明天不在，下一個人能接手嗎？
答不出來，那還不是積木，那還是你

交得出去

AI 讓寫程式變便宜了。剩下的是想法、你審查得動的深度、知道哪裡能放手哪裡不行，還有把它交得出去的能力

💡 想法 T 型 🔗 光譜 📦 交得出去

這四件事一直都在，
只是被 寫程式的繁重工作 蓋住了

| 現在 AI 把它們顯影出來而已

Q&A 謝謝 🌐 nicklecheng.turbos.tw