多奇·教育訓練

OpenAI Codex CLI 代理人開發全攻略

從入門到實戰

多奇數位創意有限公司

技術總監 黃保翕（Will 保哥）

https://blog.miniasp.com

簡介 OpenAI Codex

OpenAI 推出全新 AI 程式設計代理人

- **2025 年 5 月**，OpenAI 推出全新 AI 程式設計代理人：**Codex**
- Codex 作為 ChatGPT 的研究預覽功能，專為軟體工程任務設計
- Codex 採用 **codex-1** 模型，這是基於 OpenAI 的 o3 推理模型，經過強化學習與人類反饋(RLHF)微調，特別針對**程式設計任務**進行優化
  - Introducing OpenAI o3 and o4-mini
- Codex 能夠在**雲端沙盒環境**中執行多項任務，包括**撰寫功能**、**修復錯誤**、**執行測試**、**解釋程式碼**等
- Codex 設計強調**安全性**與**透明度**，允許用戶透過**引用文件**、**終端日誌**和**測試結果**來驗證其輸出

OpenAI Codex 包含兩個產品

- **Codex** (將**任務**委派給**雲端的軟體工程代理人**)
  - 一個**基於雲端**的**軟體工程代理** (SWE-Agent)
  - 可以同時處理多項任務，由特別訓練的 **codex-1** 模型提供支援。
  - 現已開放給 **ChatGPT Pro**、**Team** 和 **Enterprise** 用戶使用，**Plus** 用戶即將推出。
  - 目前限制非常多，價格昂貴，開發過程無法連網，還看不到原始碼，實用性有限！
  - 詳細介紹: https://openai.com/index/introducing-codex/
- **Codex CLI** (在**終端機**中與**輕量級程式設計代理**合作開發)
  - 一個**基於終端機介面**的**軟體工程代理** (SWE-Agent)
  - 採用 Apache-2.0 license 授權的**開放原始碼專案**
  - 允許你使用支援 OpenAI Chat Completions API 的其他 LLM 供應商 (包含 Azure 雲端)
  - 支援全自動審核運行模式，可與 CI 環境整合

簡介 OpenAI Codex CLI

Codex CLI 系統需求

系統需求	詳細資訊
作業系統	macOS 12+、Ubuntu 20.04+/Debian 10+ Windows 11 透過 WSL2
Node.js	22 或更新版本 (建議使用 LTS)
Git (選用) (建議安裝)	2.23+ 內建 PR 輔助工具
RAM	至少 4 GB 記憶體 (建議 8 GB)

Codex CLI 可完全在本地執行

- Codex CLI 可以完全在**本機運行**
  - 不支援 Windows 環境，但可以跑在 **WSL** 底下！
  - 其實 o3 與 o4-mini 模型就是針對 Linux 的 Shell 環境最佳化的！
- 大語言模型可以自由選擇**地端**或**雲端**模型
  - openai (預設)
  - azure
  - openrouter
  - gemini
  - ollama
  - mistral
  - deepseek
  - xai
  - groq
  - arceeai
  - 任何其他與 OpenAI API 相容的供應商

關於 Codex CLI 的特色

- **本地執行**
  - Codex CLI 完全在開發者的機器上運行，確保敏感程式碼和資料的隱私與安全
- **多模態輸入**
  - 支援文字、截圖或低保真草圖等輸入方式，提供直觀的程式設計互動
- **支援多種模型**
  - 支援 OpenAI 的最新模型，如 o3 和 o4-mini，未來計劃包括 GPT-4.1 等
- **非互動式 / CI 模式**
  - 適用於無須人為介入的工作情境（Code Review, 實作功能, 自動回答問題）
- **開放原始碼**
  - GitHub 上開源，鼓勵社群貢獻與透明發展
- **安全性**
  - **三種執行模式：建議**（預設）、**自動編輯**、**完全自動**
  - 使用**沙盒技術**（macOS 上的 Apple Seatbelt，Linux 上的 Docker）確保安全執行

Codex CLI 支援多模態輸入

- 除了基本「文字」輸入外，也可以輸入「圖片」來進行提問

```
codex -i SNAG-20250526-201621-33.png "What's in this image?"
```

- 可以支援一次提問多張圖片
  - 加入多個 **-i** 參數即可

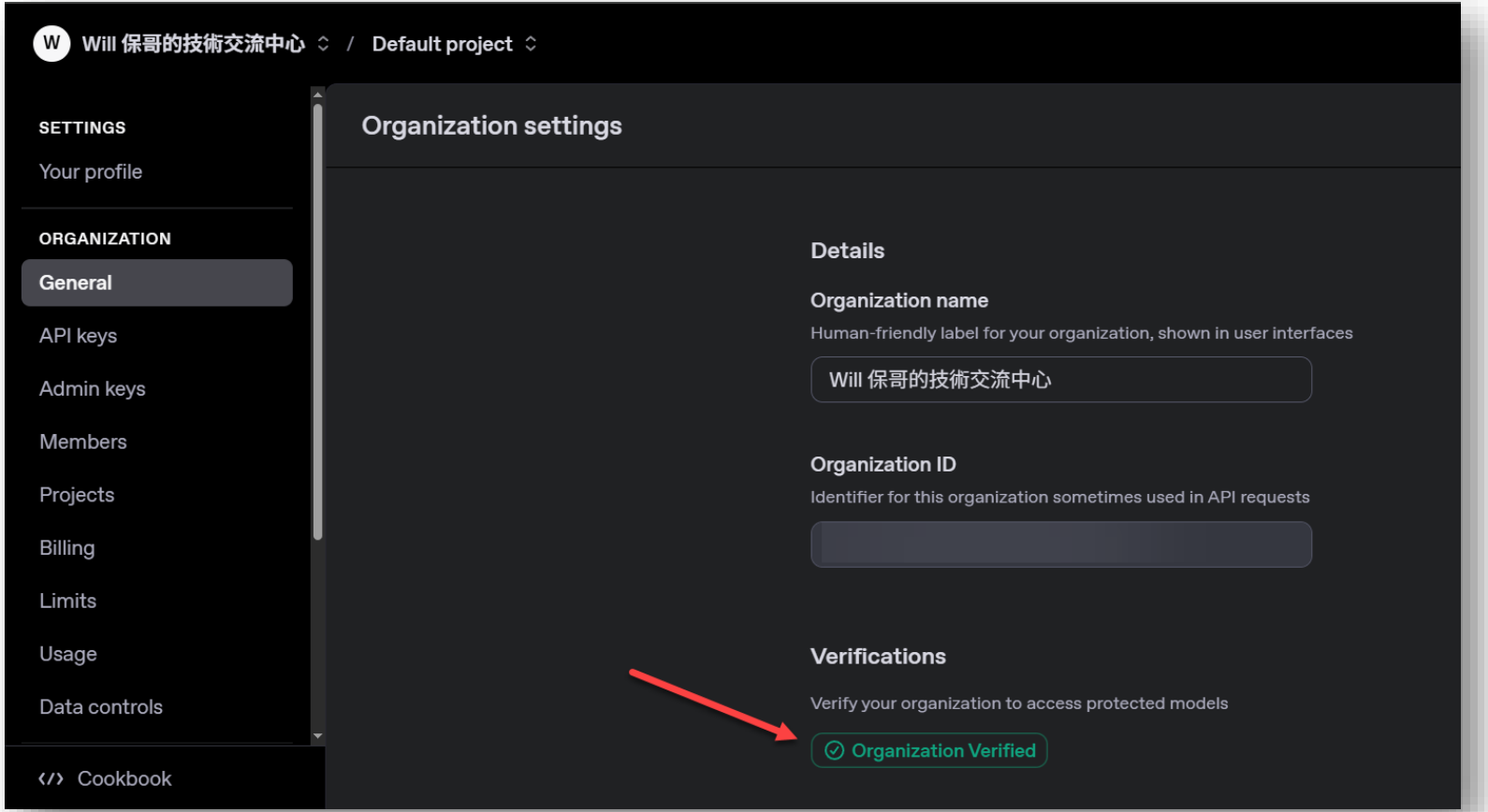
- Bug & PR
  - https://github.com/openai/codex/pull/1122

Codex CLI 支援多種模型

- **任何支援 Responses API 的模型都可以使用 Codex CLI**
  - 建議使用模型: **o4-mini** (預設模型), **o3**, **codex-mini-latest**
  - 若你使用 ChatGPT 帳號登入，預設模型會被改為 **codex-mini-latest**
  - 你可以從 Codex CLI 傳遞 **`--model gpt-4.1`** 參數來覆蓋預設值
  - 你的 API 帳號可能需要組織驗證才能開始串流回應並從 API 查看思考鏈摘要

- **任何支援 Chat Completion API 的模型提供者，也可以使用 Codex CLI**
  - 目前 **Codex CLI** 的 **azure** 提供者尚未支援 Responses API 等進階功能 (PR #1122)
  - 任何**非 openai 的模型提供者**，都可以透過 Codex CLI 運作，只是會缺乏進階功能！
  - 你可以從 **~/.codex/config.json** 設定檔設定 Codex CLI 的運作參數

使用 OpenAI 的 API 帳號可能需要組織驗證



https://platform.openai.com/settings/organization/general

簡介 Responses API

- OpenAI Responses API 於 2025 年 3 月首次推出
  - 早期的 **Chat Completions API** 已經無法滿足先進 Agentic AI 的需求
  - 全新的 **Responses API** 主要加強了**代理人**與**工具調用**的能力
  - Responses API 具有以下內建的工具能力
    - 網路搜尋工具 (WebSearchTool)、檔案搜尋功能 (FileSearchTool)
    - 電腦使用能力 (ComputerTool)、函式調用 (FunctionTool)
    - 圖片生成 (Image generation)、 程式碼執行 (Code Interpreter)、Remote MCP
- Azure OpenAI Responses API (Preview)
  - `2025-03-01-preview` (Responses API & support for computer-use-preview model) (圖片輸入)
  - `2025-04-01-preview` (**Reasoning summary** with **o3** and **o4-mini**)
  - Azure OpenAI in Azure AI Foundry Models API preview lifecycle

Codex CLI 支援非互動模式 (CI 模式)

- 適合在 CI 的 Pipelines 中運行
- 加入 **-q** 或 **--quiet** 即可進入非互動模式
- 執行時的輸出格式為 JSON Lines（ JSON-L ）
  - 每一行都是一次 API 的呼叫與回應

- 可以透過設定環境變數 **DEBUG=true** 來啟用偵錯記錄
  - 最近一次的偵錯記錄檔位置
    - `~/.local/oai-codex/codex-cli-latest.log`
  - 所有偵錯記錄檔位置： `~/.local/oai-codex/codex-cli-*.log`

開放原始碼

- GitHub Repo: https://github.com/openai/codex
  - 目前的 Node.js 版本比較完整，另有一個 Rust 版本正在緩慢開發中
- 我已經發了 8 個 PR 了，沒有一個被合併成功，持續努力！😅
  - #479 Add a PowerShell script that can better support Windows (支援 Windows)
  - #1004 The log file name should not use the colon character (支援 Windows)
  - #1121 codex -v <rollout> is not working (CLI 命令用法的 Bug)
  - #1122 fix: Support Azure OpenAI Responses API (為了讓 AOAI 支援圖片)
  - #1125 Allow running without a sandbox if the user has explicitly marked the environment as already being sufficiently locked-down.
  - #1130 docs: Clarify project doc discovery and merging logic in documentation
  - #1134 feat: Add full bash completion for codex
  - #1143 fix: yq doesn't need a -o=json argument

安全性

- Codex 讓你決定代理收到的**自主權限範圍**以及**自動核准政策**
  - 透過 **`--approval-mode`** 旗標（ **`-a`** ）可以設定**自動核准政策**
    - **建議模式**: `codex -a suggest "build it and fix all issues"`
    - **自動編輯**: `codex -a auto-edit "build it and fix all issues"`
    - **完全自動**: `codex -a full-auto "build it and fix all issues"`
- 使用**沙盒技術** (macOS 上的 Apple Seatbelt，Linux 上的 Docker)
  - 預設在**完全自動**模式下，會有嚴格的使用限制：
    - 每個指令都會以**停用網路**的方式執行
    - 執行時只有**目前的工作目錄** (以及**暫存資料夾**) 可以寫入檔案，以強化防禦
  - 以**自動編輯**或**完全自動**模式啟動時, 若該目錄未使用 Git 版控會告警並需確認！

Codex CLI 提供三種核准政策

- **建議模式** (預設)
  - 自動**允許讀取**專案中任何檔案
  - 任何**編輯檔案**與**執行外部工具**(除了讀檔的命令外)都需要人工批准
- **自動編輯（--auto-edit）**
  - 自動**允許讀取**並且**對檔案進行 apply-patch 寫入**
  - **執行任何 shell 命令**都需要**人工批准**
- **完全自動（--full-auto）**
  - 自動**允許讀取**並且**對檔案進行 apply-patch 寫入**
  - 自動**允許任意 shell 命令執行** (但會**停用網路**與**限制僅工作目錄可以寫入檔案**)
  - 💡 **未來**能夠將**特定指令**加入**白名單**, 讓其在**啟用網路**的狀態下自動執行。

沙盒技術的細節

- **macOS 12+** 指令會以 Apple Seatbelt 包裹 (sandbox-exec)
  - 除了一小部分可寫入的根目錄 (**$PWD**, **$TMPDIR**, **~/.codex** 等)， 其餘全部都被放在**唯讀監獄**中。
  - **出站網路**預設會被完全阻擋， 即使子程序嘗試用 **curl** 連線到某處也會失敗。

- **Linux** - 預設沒有沙箱機制，但 OpenAI 正在發展自己的沙箱方案 (Landlock)
  - OpenAI 建議暫時用 Docker 來進行沙箱隔離， Codex 會在最小化的容器映像檔內自我啟動， 並將你的 Repo 以**讀寫**方式**掛載**到相同路徑。
  - 自訂的 **iptables/ipset** 防火牆腳本會拒絕所有外部連線，僅允許 OpenAI API 連線
  - 這能讓你在不需要主機 **root** 權限下， 獲得可預期且可重現的執行結果。
  - Bug/PR: https://github.com/openai/codex/pull/1125

示範如何打造專屬的 AI 代理人

展示步驟

- **打造隔離的開發環境**
  - **實作環境手冊**已發佈在課程頁面中！
  - 透過 Development Containers 技術，快速打造 Linux 開發環境
  - 完整的學習手冊在此: Developing inside a Container
- **與代理人共同開發專案**
  - 打造個人用的程式設計助理，隨時將手邊的工作交出去做
  - 將開發環境與代理人的環境整合在一起，兩人一起在同一個專案協作
- **與代理人各自開發專案**
  - 打造隔離的開發環境，讓代理人自行完成任務，並透過 Git 提交變更回來
  - 需要一個不錯的通知管道 (Telegram Bot)

Codex CLI 命令列參考

指令	目的	範例
`codex`	互動式 REPL	`codex`
`codex "..."`	互動式 REPL 的初始提示	`codex "fix lint errors"`
`codex -q "..."`	非互動式 CI 模式	`codex -q "explain utils.ts"`
`codex -c`	編輯使用者指令檔	`codex -c`
`codex --free`	取得 OpenAI API 金鑰	`codex --free`
`codex --auto-edit`	自動編輯模式	`codex --auto-edit "fix build warnings"`
`codex --full-auto`	自動開發模式 (沙盒環境)	`codex --full-auto "fix build warnings"`

關於「系統提示」的文件載入順序

1. prefix
   - codex-cli/src/utils/agent/agent-loop.ts#L1612-L1658
2. dynamicLines
   - codex-cli/src/utils/agent/agent-loop.ts#L1602-L1611
3. 使用者自定義提示
   - **`~/.codex/instructions.md`**
4. 執行時透過 **`--project-doc <filename>`** 載入的提示文件
   - 跟順位 3 的內容會以 "**`--- project-doc ---`**" 分隔

關於「使用者提示」的內容載入順序

- 對話歷史
  - **歷史工作階段 (唯讀)**
    - `codex --history`
    - `codex --view ~/.codex/sessions/rollout-xxx.json`
  - **接續工作階段 (可以接續提問)**
    - `codex --history`
    - 按下一個 `tab` 鍵
    - 按下 `Enter` 繼續
- **使用者提示**
  - `codex -q "user-prompt-here"`

```python
from openai import OpenAI
client = OpenAI()

response = client.responses.create(
    model="gpt-4o",
    instructions="請以正式新聞報導口吻回覆，重點放在事實。",
    input=[
        {"role": "user", "content": "介紹一下最新的太空探索計畫"}
    ]
)
print(response.output_text)
```

優化與代理人之間的互動方式

- **透過終端機執行命令**
  - 發送 Telegram 通知 (免費)
    - 在 Telegram 搜尋 **`@BotFather`**
    - 用 /start 開始建立一個 Bot
    - 取得 Bot Token
    - 完整的 Telegram Bot API 文件: https://core.telegram.org/bots/api
- **透過 VS Code 的 Tasks 呼叫 codex 執行**
  - Integrate with External Tools via Tasks
  - 建立 **`.vscode/tasks.json`** 設定檔
  - 把常見的工作與任務寫成 shell script 可以更加精準的完成工作

深入講解 OpenAI Codex CLI 的提示工程

記憶能力與專案文件

- **持續維護上下文**
  - **~/.codex/instructions.md**
    - 個人全域提示
  - **AGENTS.md**
    - 優先找到**目前工作目錄**的 **AGENTS.md**
    - 如果找不到，就會往上找到 **repo 根目錄**的 **AGENTS.md** 文件
- **加入自訂的專案文件**
  - 使用 `--project-doc <file>` 即可，使用此參數就不會載入 **AGENTS.md**
- **除專案文件或記憶**
  - 使用 **--no-project-doc** 參數，或設定環境變數 **CODEX_DISABLE_PROJECT_DOC=1**
  - 可以藉此停用記憶能力，不會載入任何專案文件

~/.codex/instructions.md

- 可以透過命令列開啟 vim 編輯
  - **`codex -r`**
- 可以透過 Visual Studio Code 開啟編輯
  - **`code -r ~/.codex/instructions.md`**
- 可以寫個人全域提示 (請用 Markdown 文件)
  - **use zh-tw**
- 可參考 codex 的系統提示，不用定義重複的內容
  - https://github.com/openai/codex/blob/6b5b184f21504c99eeeee56b9d607eb8576f96db/codex-cli/src/utils/agent/agent-loop.ts#L1602-L1659

AGENTS.md

- 建議統一寫在 Repo 根目錄，並將該檔案加入版控
  - 高階架構說明
  - 目錄結構與用途
  - 主要的服務與工具類別
  - 資料庫結構的參考路徑
- 額外的專案文件
  - 可以使用 **`--project-doc <file>`** 載入文件當上下文
  - 但須注意，加入自訂的專案文件就不會載入 **`AGENTS.md`** 文件
  - 使用範例

```
codex --project-doc .github/copilot-instructions.md
```

交辦工作的技巧

- 要叫代理人做的工作，不要寫在 **AGENTS.md** 裡面

- 如果要，那就定義一些「章節」讓使用 codex 時呼叫，例如：
  - **Follow the "Task 1" instructions defined in project-doc section.**

- 載入大量使用者提示給 codex 的方法
  - 先準備好一個文字檔 **/tmp/PROMPT.md**
  - 透過以下命令執行 codex

```
codex -q "$(cat /tmp/PROMPT.md)"
```

寫入大量文字的兩種 shell script 語法

支援變數內插指令語法

```
cat <<EOF > /tmp/PROMPT.md
Current time: $(date '+%Y/%M/%d %H:%M:%S')
Current user: $(whoami)
Current working directory: $(pwd)

# Prompt
You are a helpful AI assistant. Very long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long text
EOF
```

關閉變數內插指令語法

```
cat <<'EOF' > /tmp/PROMPT.md
# Prompt
You are a helpful AI assistant. Very long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long long
  long long long long long long long text
EOF
```

vibe coding

- 幫我寫 utils/date.ts 的單元測試
- 尋找漏洞並建立安全性檢視報告
- 仔細審核此專案，並提出3個具有高影響力且範圍明確的PR。
- 解釋這個正則表達式的作用：^(?=.*[A-Z]).{8,}
- 使用 git mv 批次重命名 *.jpeg 為 *.jpg
- 找出整個專案所有用到英文大寫的副檔名，並幫我全部改為英文小寫
- 將 Dashboard 元件重構為 React Hooks
- 更新 CHANGELOG 準備下次發行

其他有趣的玩法

- **替代理人取名**
  - 不一定只能叫 Codex，以後可用代名詞稱呼，例如: Jarvis
  - 直接抓原始碼回來，並且直接修改系統提示
    - 檔案路徑: **codex-cli/src/utils/agent/agent-loop.ts**

- **與 GitHub Copilot / Cursor / Windsurf / ... 共用提示文件**
  - 常見的 Prompts 與 Instructions 都可以透過 shell script 來動態合併多檔

詳細估算 OpenAI Codex CLI 的導入成本

OpenAI API (incl. Azure) 的 Tokens 費用

模型名稱	輸入費用 (USD/1M Tokens)	輸出費用 (USD/1M Tokens)	快取輸入費用 (USD/1M Tokens)	上下文視窗 (Tokens)
o3	$10.00	$40.00	$2.50	200K
o3-mini	$1.10	$4.40	$0.55 (Azure)	200K
o4-mini	$1.10	$4.40	$0.28 (Azure)	200K
codex-mini-latest	$1.50	$6.00	適用提示快取 (75% 折扣)	200K
gpt-4.1	$2	$8	$0.5	1M

誰該支付「代理人」的出場費 (Tokens)

- **個人用的代理**
  - **提升在公司的表現** (你是否要將**不擅長**的部分外包做一部分？)
  - **加速自己當前工作** (你是否要將**最緊急**的工作外包做一部分？)
  - **逃避不想做的工作** (你是否要將**不想做**的工作外包做一部分？)

- **公司用的代理**
  - **加速整體工作效率** (公司是否希望可以提升所有人的工作效率？)
  - **提升客戶服務表現** (公司是否希望能透過代理人提升軟體品質？)
  - **降低成本縮短工時** (公司是否希望能夠減少大家的工作時數？)

聯絡資訊

The Will Will Web

網路世界的學習心得與技術分享

http://blog.miniasp.com/

Facebook

Will 保哥的技術交流中心

http://www.facebook.com/will.fans

Twitter

https://twitter.com/Will_Huang

多奇·教育訓練

THANK YOU!

Q&A