

課程內容詢問



這堂課在幹嘛？

你是指哪一堂課呢？😄

可以跟我說課名或課程簡介嗎？這樣我才能幫你解釋它在上什麼、主要學什麼內容。



Learn to build and host applications with an AI agent in Vibe Coding 101 with Replit, built in partnership with Replit and taught by Michele Catasta, President, and Matt Palmer, Head of Developer Relations.

Coding agents are changing how we write code. “Vibe coding” is a growing practice where you lean into letting a coding agent do most of the heavy lifting while you focus on the architecture and features of your application. But effective vibe coding isn’t just about one-shot prompting, accepting all recommendations, and hoping for the best. It involves structuring your work, refining your prompts, and using frameworks that lead to cleaner, more efficient code.

In this course, you’ll use Replit’s cloud environment— with an integrated code editor, package manager, and deployment tools—to build and deploy two web applications with the help of an AI coding agent. Along the way, you’ll learn strategies for working effectively with agents and improve your development skills in the process.

In detail, you’ll:

Understand principles of agentic code development such as being precise, giving agents one task at a time, making prompts specific, keeping projects tidy, starting with fresh sessions for each new feature, and adopting patience while debugging.

Learn how to get started with Replit, and understand fundamental concepts behind effective AI collaboration including a “five-skill framework” in vibe coding of thinking, using frameworks, checkpoints, debugging, and providing context.

Create a product requirement document (PRD) and wireframe for your agent to build a prototype of a website performance analyzer.

Use the agent to make your prototype more visually appealing, switch to assistant mode to customize your applications with screenshots and prompts, and deploy your application so others can access it online.

Build a head-to-head national park ranking app, from a sample dataset, with voting capabilities and persistent data storage.

Refine your app by integrating a complete dataset, and ask the assistant to recap and explain what it built to find room for improvement and reinforce your learning.

By the end of this course, you’ll have a solid foundation in building with coding agents—and a process you can use to keep vibe coding effectively.

這門課其實是在教你 **怎麼跟 AI 程式代理人 (coding agent) 合作**，把應用程式建出來並上線，而不是自己從零到一行行寫程式。簡單講，它不是單純的「學寫程式」，而是學 **如何有效使用 AI 寫程式**。

課程重點大概可以拆成幾塊：

1. 觀念與方法

- 學習「vibe coding」的心法：把 AI 當成主力工程師，你是架構師。
- 怎麼下清楚的任務、分解問題、一次只給 AI 一件事。
- 學會 debug、檢查 AI 輸出的程式碼，不是全盤接受。
- 建立一個系統性的合作流程（五技能框架：思考、用框架、檢查點、除錯、提供上下文）。

2. 實作環境

- 用 **Replit**（一個雲端 IDE）來寫程式、管理套件、部署應用程式。
- 讓你可以快速從 AI 生成程式 → 修改 → 直接上線給別人用。

3. 專案練習

- **第一個專案**：做一個網站效能分析工具。
 - 先寫產品需求文件 (PRD)、畫 wireframe，讓 AI 幫你生 prototype。
 - 再用 AI 美化、加功能、最後部署上線。
- **第二個專案**：做一個「國家公園排名投票網站」。
 - 有投票、資料存取功能。
 - 後面再用完整資料集擴充，請 AI 幫忙總結、解釋程式碼，學怎麼回顧與改進。

4. 學到的能力

- 跟 AI 協作開發應用程式的完整流程。
- 怎麼把一個想法變成可用的線上網站。
- 增強自己的軟體開發思維（規劃、分解問題、驗證功能），而不是單純的語法練習。

👉 簡單一句話：

這堂課教你 **用 AI 寫程式 & 上線**，你主要負責 **想法、架構、檢查**，AI 幫你處理大部分程式碼。

要不要我幫你用一張「課程地圖」把它畫出來，讓你更快看懂流程？



我該如何使用 vibe coding 的方式寫一個 javascript 的小遊戲。

很好問題 👍！你要用 **vibe coding** 的方式做一個 JavaScript 小遊戲，可以把流程想像成：**你設計 → AI 寫主要程式 → 你檢查 + 調整**。

這裡我幫你整理成一個 vibe coding 的步驟藍圖：

▯ Vibe Coding 寫 JS 小遊戲的流程

1. 定義遊戲想法 (PRD 簡易版)

先用白話文告訴 AI：

- 遊戲是什麼？（例：打磚塊、猜數字、跑酷）
- 玩家要怎麼操作？（鍵盤、滑鼠、點擊）
- 勝負條件是什麼？（得分、闖關、時間限制）

👉 範例 PRD：

「我要做一個 JavaScript 小遊戲，類似打磚塊（Breakout），玩家用滑鼠左右移動板子反彈球，打掉所有磚塊就贏。希望畫面用 HTML Canvas，控制方式簡單，不需要太多花俏設計。」

2. 拆解任務，逐步請 AI 幫忙

不要一次叫 AI 生完整遊戲，而是分步驟：

1. 先生成基本畫布 + 遊戲主迴圈。
2. 再加「球的移動」邏輯。
3. 接著加「板子」和控制。
4. 再加入「磚塊排列」。
5. 最後加上「碰撞檢測」+ 計分。

👉 每步結束後，自己執行程式、看效果，再決定下一步。

3. 給 AI 的 prompt 範例

你可以這樣問 AI：

- 「請幫我用 HTML + JavaScript 建立一個遊戲畫布，顯示一個會自動移動的小球。」
 - （測試沒問題後）「再加一個可左右移動的板子，讓玩家用左右方向鍵控制。」
 - 「加上磚塊牆，排列成 5 列，每列 10 個磚塊。」
 - 「讓球能打掉磚塊，並加上分數計算。」
-

4. Debug 與美化

- 如果出現 bug，先把錯誤訊息貼給 AI，請它修。
 - 想要換顏色、速度、字型，也可以直接問 AI：「讓背景變黑，球變紅色，板子變藍色。」
-

5. 部署與分享

完成後：

- 可以直接在 **Replit** 建專案，貼上 HTML/JS。
 - 點擊「Run」測試，然後用 Replit 的 **Deploy** 把遊戲上傳，分享網址給朋友玩。
-

⚡ 總結：

用 vibe coding 寫小遊戲就是：

1. 先寫簡單需求（PRD）。
2. 把任務拆小，逐步叫 AI 寫。
3. 你專注在檢查、測試、改需求，AI 寫大部分程式。

要不要我幫你示範 **第一步 prompt + 對應程式碼**（小球自動移動的版本）給你看，讓你直接能跑？



我要做一個 JavaScript 小遊戲，類似打磚塊（Breakout），玩家用方向箭頭左右移動板子反彈球，打掉一個磚塊加一分，打掉三十個磚塊就獲勝結束遊戲，球掉到底下就失敗可以重新開始。希望畫面用 HTML Canvas，控制方式簡單，不需要太多花俏設計。

```

<!--
Breakout (30-brick win) - Vibe Coding Example
- Simple HTML + JavaScript single-file game using <canvas>
- Controls: Left / Right arrow keys
- Rules: Each brick destroyed = +1 score. Destroy all 30 bricks to win.
- If ball falls below paddle -> lose. Press Space or click Restart to play again.

How to use:
- Copy this file into Replit (new HTML/CSS/JS repl) or save as breakout.html and open in browser
- If you want me to walk through generating this step-by-step with an AI agent, I can show prompt
-->

<!doctype html>
<html lang="zh-Hant">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>Breakout – Vibe Coding Example</title>
  <style>
    html,body{height:100%;margin:0;font-family:system-ui,Segoe UI,Roboto,"Noto Sans",sans-serif;
    #gameCanvas{background:#0b1220;border:6px solid #1f2937;border-radius:8px;display:block};
    .note{position:fixed;left:16px;bottom:12px;font-size:13px;color:#9aa4b2}.
  </style>
</head>
<body>
  <canvas id="gameCanvas" width="720" height="480"></canvas>
  <div class="note">Controls: ← → arrows. Press <strong>Space</strong> to restart after game over

  <script>
    // --- Game config ---
    const canvas = document.getElementById('gameCanvas');
    const ctx = canvas.getContext('2d');

    const PADDLE_WIDTH = 100;
    const PADDLE_HEIGHT = 14;
    const PADDLE_Y_OFFSET = 30; // distance from bottom

```


我已把完整的 **HTML + JavaScript 小遊戲 (Breakout)** 放到畫布上，檔名：「Breakout - Vibe Coding Example (HTML)」。

使用說明（簡短）：

- 你可以直接在 Replit 建一個新的 HTML 專案，或把檔案另存為 **breakout.html**，然後在瀏覽器開啟就能玩。
- 操作：使用 **左右方向鍵** 控制板子；打掉 30 個磚塊就贏；球掉到畫面底部則失敗，可按 **Space** 或點擊畫布重新開始。

你想要我接下來幫你做哪一件事？

- 將這個遊戲分步用 **AI agent 生成的 prompt 範例**（做成 vibe coding 流程）？
- 或是幫你**美化 UI / 加音效 / 加行動裝置觸控支援 / 加排行榜**？哪個優先？



我該如何在vs code 執行此程式

在 **VS Code** 執行這個 HTML + JavaScript 遊戲很簡單，因為它不需要額外安裝套件，只要用瀏覽器打開就能跑。步驟如下：

🔧 方法 1：直接開啟 HTML

1. 在電腦新建一個資料夾，例如 **breakout-game**。
2. 把我給你的程式碼存成 **index.html** 放進去。
3. 在 VS Code 打開這個資料夾。
4. 右鍵 **index.html** → 選擇「**Open with Live Server**」（需要安裝 **Live Server 擴充套件**）。
 - 如果沒安裝，可以到 VS Code 的 Extensions 搜尋 **Live Server** 安裝。
5. 瀏覽器會自動開啟網址（通常是 **http://127.0.0.1:5500/index.html**），遊戲就能玩了。

🔧 方法 2：手動用瀏覽器開啟

1. 同樣把程式碼存成 **index.html**。
2. 在檔案總管裡雙擊它，或右鍵 → 用 Chrome / Edge / Firefox 打開。
3. 遊戲也能正常執行。

🔧 方法 3：內建簡易伺服器（可選）

如果你不想裝 Live Server，也可以用 VS Code 的 **內建終端機**，執行一個簡單的伺服器：

- **Python 版本**（如果電腦有 Python）：

```
bash  
  
python -m http.server
```

然後在瀏覽器開啟 `http://localhost:8000/index.html`。

- **Node.js 版本**（如果有 Node.js）：

```
bash  
  
npm run serve
```

也會在瀏覽器提供一個網址。

👉 建議用 **方法 1（Live Server）**，最簡單，適合初學者，而且每次存檔會自動重新整理頁面。

要不要我幫你做一個 **超精簡操作步驟清單（含截圖流程）**，讓你第一次設定 VS Code 就能立刻執行遊戲？



要怎麼把程式碼存成 `index.html` 放進去

好 🍌 我來用最簡單的步驟告訴你怎麼做，把我給你的程式存成 **index.html** 然後放進 VS Code：

📝 存成 `index.html` 的方法

1. 開資料夾

- 在桌面上新建一個資料夾，例如叫做 **breakout-game**。

2. 開啟 VS Code

- 打開 VS Code → 上方選單 **File** → **Open Folder**（檔案 → 開啟資料夾）。
- 選剛剛建立的 **breakout-game** 資料夾。

3. 新增檔案

- 在 VS Code 左側檔案總管，按右鍵 → **New File**（新增檔案）。
- 輸入檔名：`index.html`。

4. 貼上程式碼

- 打開 `index.html`，把我之前提供的整份程式碼貼進去。
- 然後 **存檔 (Ctrl+S)**。

5. 執行

- 如果已經裝了 **Live Server** 擴充套件：
 - 在 `index.html` 上按右鍵 → **Open with Live Server**。
 - 瀏覽器會自動打開遊戲。

- 如果沒有安裝，也可以直接去檔案總管，雙擊 `index.html` → 它會在瀏覽器打開，遊戲就能跑了。
-

這樣遊戲就能在 VS Code（或瀏覽器）執行 🎮。

要不要我幫你畫一個「步驟圖解流程（含截圖示意）」讓你第一次操作就完全不會卡住？