

## CHAPTER 08

# 前端框架與現代網站開發

HORAZON

互動媒體設計

# 網頁發展史：從靜態到動態

1. **靜態網頁 (Static)**：純 HTML/CSS，內容寫死。
2. **動態網頁 (Dynamic)**：Server 產生 HTML (PHP, ASP.NET)，換頁要重載。
3. **單頁應用 (SPA)**：前端負責跟 Server 要資料，**只有局部更新**。

**關鍵技術：AJAX**

# 什麼是 AJAX?

- Asynchronous JavaScript And XML
- (現在大部分是傳送 JSON 格式，不是 XML 了)

**重點：**

**網頁不需要重新整理**，就可以跟伺服器 (Server) 拿資料並更新畫面。

想像你去餐廳：

**傳統網頁：**加點飲料，服務生把整張桌子收走，再重新擺盤一次。

**AJAX：**服務生直接把飲料端上來，原本的菜不動。

# AJAX 現代寫法：FETCH API

以前用 `XMLHttpRequest` (很難寫)，現在用 `fetch`。

```
// 向伺服器要資料
fetch('https://api.example.com/data')
  .then(response => response.json()) // 把回應轉成 JSON
  .then(data => {
    console.log(data); // 拿到資料了！
    document.querySelector("#box").innerText = data.name;
  });
```

這就是現代網頁 (Facebook, Gmail) 順暢的原因。

# 現代前端框架 (FRONTEND FRAMEWORKS)

隨著網頁越來越複雜，純寫 JavaScript (Vanilla JS) 變得難以維護。  
於是誕生了三大框架：

1. **Vue.js** (好學、靈活、台灣很紅) ●
2. **React** (Facebook維護、生態系最大) ●
3. **Angular** (Google維護、架構嚴謹) ●

它們的共同點：**組件化 (Component)** 與 **資料驅動 (Data-Driven)**。

# 為什麼選 VUE.JS?

- 直覺：HTML/CSS/JS 寫在一起，很像原本的網頁開發方式。
- 雙向綁定：資料變了，畫面自己會變 (不用一直寫 `document.querySelector`)。

```
<script setup>
import { ref } from 'vue'
const count = ref(0) // 定義變數
</script>

<template>
  <h1>目前數字：{{ count }}</h1> <!-- 自動更新 -->
  <button @click="count++">+1</button> <!-- 事件 -->
</template>
```

1. **AJAX** 讓網頁不用重整就能更新資料 (Fetch API)。
2. **Vue.js** 等框架讓開發大型網頁變簡單 (資料驅動)。
3. 現代網頁開發強調**組件化**與**使用者體驗**。