補充教材

資料儲存：TINYDB

HORAZON

應用程式設計

本章目標

1. 理解 **資料儲存 (Data Persistence)** 的重要性
2. 認識 **TinyDB (微型資料庫)**
3. 實作：製作具有記憶功能的記分板
4. 實作：簡易備忘錄 (Memo)

為什麼需要資料儲存？

問題情境

玩遊戲玩到一半，**關掉 App 再打開**...
分數歸零了？設定跑掉了？
這樣的使用者體驗非常糟糕！

解決方案

我們需要把資料存在手機的**硬碟 (Storage)** 裡，而不是記憶體 (RAM) 裡。
即使關機重開，資料依然存在。這就叫做**資料持久化 (Persistence)**。

認識 TINYDB

Run Time (執行時) 的資料儲存元件。

- **Key-Value (鍵值對)** 儲存格式
- 就像是一個超大的**置物櫃**
- 每個格子都有一個**標籤 (Tag)**
- 透過標籤來存取資料

常用積木

- `StoreValue (tag, value)`: 存資料
- `GetValue (tag, valueIfTagNotThere)`: 讀資料

實作 1：永久記分板

製作一個可以記錄最高分的按鈕。

1. 畫面設計

- **Label_Score**: 顯示目前分數
- **Button_Click**: 按下去 +1 分
- **Label_HighScore**: 顯示最高分 (要存起來！)
- **TinyDB1**: 記得拉進去 (在 Storage 分類下)

實作 1：程式邏輯 (存)

當分數改變時，檢查是否破紀錄。

```
當 Button_Click.Click
    set global score to (get global score + 1)
    Label_Score.Text = score

    // 檢查是否破紀錄
    如果 (score > TinyDB1.GetValue("HighScore", 0))
        TinyDB1.StoreValue("HighScore", score)
        Label_HighScore.Text = score
```

實作 1：程式邏輯 (讀)

當 App 啟動時 (Screen.Initialize)，要把舊紀錄讀回來。

```
當 Screen1.Initialize
    // 讀取最高分，如果沒存過就給 0
    set global highScore to TinyDB1.GetValue("HighScore", 0)
    Label_HighScore.Text = highScore
```

這樣不管重開幾次 App，最高分都會留著！

實作 2：簡易備忘錄

讓使用者輸入文字並存起來。

畫面設計

- TextBox: 輸入備忘事項
- Button_Save: 儲存按鈕
- Label_Memo: 顯示儲存的內容

程式邏輯

- 存：`TinyDB1.StoreValue("MyMemo", TextBox.Text)`
- 讀：`Label_Memo.Text = TinyDB1.GetValue("MyMemo", "沒有資料")`

TINYDB 的限制

- **適用範圍**：簡單的設定、分數、少量文字。
- **不適用**：
  - 大量資料 (例如幾千筆訂單) -> 請用 SQL
  - 需聯網共享的資料 -> 請用 CloudDB 或 Firebase
  - 圖片或影片 -> 請存檔案路徑

但在單機小遊戲中，TinyDB 是最強大的夥伴！

進階：CLOUDDB (雲端資料庫)

如果想讓**不同手機**的使用者也能看到同樣的資料 (例如：全伺服器排行榜、聊天室)，就要用 **CloudDB**。

CLOUDDB 特點

- 資料存在網路上 (Redis Server)。
- 即時更新 (Real-time)。
- **DataChanged** 事件：當資料改變時，所有連線的手機都會收到通知。

適用情境

- 多人連線遊戲
- 即時聊天室
- 投票系統

重點回顧

- TinyDB 用於手機本機儲存資料。
- 使用 Tag (**標籤**) 來區分不同的資料。
- 記得處理 Screen.Initialize (初始化) 來讀取舊資料。
- 資料儲存是讓 App 變得「**有用**」的關鍵一步！

下一章：期末專案衝刺！

未來展望：從積木到代碼

App Inventor 是一個很棒的起點，但如果你想成為專業的 App 開發者，下一步是什麼？

1. 跨平台開發 (推薦)

一次寫好，同時在 Android 和 iOS 執行。

- **Flutter**: 使用 Dart 語言 (Google 開發，目前最熱門)。
- **React Native**: 使用 JavaScript (Facebook 開發)。

2. 原生開發 (NATIVE)

針對特定平台進行極致優化。

- **Android**: 學習 **Kotlin** 或 **Java**。
- **iOS**: 學習 **Swift**。

不要害怕程式碼，邏輯都是一樣的！

下一章：期末專案衝刺！