

CHAPTER 10

遊戲管理與核心邏輯 (SINGLETON) HORIZON

手遊程式設計

章節目標

- Trigger Event：親手寫出吃金幣邏輯
- Singleton 模式：製作 GameManager
- UI 更新：簡易的分數顯示

1. 再次複習 TRIGGER

在 Phase 1 (Ch05) 我們用過現成的金幣。現在我們要自己造一個。

1. 場景放一個金幣 Sprite。
2. 加入 **Circle Collider 2D**，勾選 **Is Trigger**。
3. **重要**：在 Tag 欄位 (Inspector 最上面) 選擇 **Add Tag** -> 新增 **Coin** -> 把金幣 Tag 設為 **Coin**。

2. 撰寫互動腳本

在 `PlayerController` 加入以下函式 (與 `Start` / `Update` 同層級)：

```
// 當進入 Trigger 感應區時會自動執行
void OnTriggerEnter2D(Collider2D other)
{
    // 如果碰到東西的標籤是 Coin
    if (other.tag == "Coin")
    {
        Debug.Log("吃到金幣了！");
        Destroy(other.gameObject); // 銷毀碰到的那個金幣
    }
}
```

存檔試玩，金幣應該會消失，且 Console 會顯示訊息。

3. 誰來記分？ (GAMEMANAGER)

如果把「分數」變數寫在主角身上，萬一主角死掉(被銷毀)，分數就不見了！
我們需要一個**永遠存在**的管理者：GameManager。

1. 新增空物件命名為 `GameManager`。
2. 建立腳本 `GameManager.cs`。

4. 單例模式 (SINGLETON)

這是一個進階但必學的寫法，讓所有程式都能輕易找到 GameManager。

```
public class GameManager : MonoBehaviour
{
    public static GameManager instance; // 靜態實例 (唯一的)
    public int score = 0; // 分數

    void Awake()
    {
        instance = this; // 遊戲開始時，把自己填入這個欄位
    }

    public void AddScore(int amount)
    {
        score += amount;
        Debug.Log("目前分數: " + score);
    }
}
```

5. 串接：主角呼叫經理

修改主角的 `OnTriggerEnter2D`：

```
if (other.tag == "Coin")
{
    // 呼叫 GameManager 的 instance 來加分
    GameManager.instance.AddScore(1);

    Destroy(other.gameObject);
}
```

解析：主角不需要知道 GameManager 在哪裡，只要透過 `instance` 就能找到它。這是 Singleton 最強大的地方。

6. (預告) 陷阱與受傷

同樣的邏輯，我們可以製作陷阱：

1. 設定陷阱 Tag 為 `Trap`。
2. 互動邏輯：

```
else if (other.tag == "Trap")
{
    // 重新讀取關卡（死掉重來）
    // 需引用 using UnityEngine.SceneManagement;
    SceneManager.LoadScene(SceneManager.GetActiveScene().name);
}
```


總結 (PHASE 2 完成)

恭喜！你已經完成了一個真正的遊戲核心：

- 角色能跑能跳 (Input, Physics)。
- 角色有動畫 (Animator)。
- 能吃金幣並記分 (Trigger, Singleton)。
- 碰到陷阱會重來 (SceneManager)。

下一章 Phase 3，我們要把它包裝得更像商品，加上 UI 與 音效！