重現 Flappy Bird

1. Flappy Bird 介紹

2013 年，一款名為 Flappy Bird 的手機遊戲橫空出世，憑藉著簡單的操作與各種天時地利人和，稱霸了 2014 年幾乎全球的手機遊戲市場。然而，在各種因素下，作者選擇在之後下架這款遊戲，使這份經典成為回憶。然而這樣簡單好上手的小遊戲，正好適合我 Unity 的入門作，因此我想要透過現存於網路上的 Flappy Bird 遊戲貼圖與 C sharp 和 Unity 遊戲引擎，重現這個曾經風靡一時的經典。

2. 基礎遊戲機制

Flappy Bird 的基礎遊戲機制十分地簡單，操控一隻小鳥，避開水管，獲取更高的分數，一旦接觸到水管便會遊戲結束，因此便要注重在遊戲物件碰撞的設計。

3. 主要問題：效能

製作這份簡單的小遊戲總共花費了大約兩天的時間，卻有將近一半的時間是在解決隱患：暫時不存在的效能問題。Unity 引擎運行的效能很大一部份與遊戲的 frame rate(影格速率)相關，遊戲運行時的許多功能都是以 frame rate 的頻率運行；但是當單一畫面的負擔過大，整個遊戲的流暢度便會降低。

我的遊戲中判定結束與計算分數的方法，是檢測玩家的碰撞範圍接觸指定區域後，依照接觸到的 tag 分別執行 GameOver 或 IncreaseScore；然而這兩份副

程式是在另一份檔案 GameManager 裡面，經驗不多的我最初選擇了 FindObjectOfType 的方式存取，但這樣的作法是檢測 GameManager 中所有的副程式中是否有存在上述兩者再執行。這種遍歷檢測的方式對效能的負擔極大，因為這個遊戲的機制非常簡單，規模也不大，倖免於卡頓等問題，不過維持這樣的寫法肯定不是長久之計。

4. 單例化/Singleton

在查閱相關 API 文件後，我發現了有兩種較為合適的解決方案，分別是制定 Event System 與 Singleton(單例化)，由於 Event System 已經練習過，這次我選擇使用 Singleton。單例化可以理解為欲呼叫的程式建立一條專屬的通道，可以省去反覆搜尋整個檔案中是否存在該副程式的時間與效能。(詳細學習資料於附錄中)

5. 音效

Singleton 的概念也可以延伸至其他範圍，如遊戲音效；透過創建 AudioManager 以搭建音效檔案與 Player 的通道，也能夠簡單地為作品增加音效。

6. 心得與總結

如果一個檔案中只有一處呼叫該副程式，可選擇使用單例化以減少運行負擔，並將原先使用 FindObjectOfType 移至 Awake(於啟動時執行)使其只會執行一次。雖然這樣的改動對我的結果影響微乎其微，但這是以遊戲規模為前提，如

果規模放大，機制稍微複雜化，便會有卡頓等情形產生。

總而言之，這樣的修改是在奠定之後進行更大專案實的良好習慣，也增強了對物件導向與遊戲設計的觀念。

7. 附錄：

檔案位置：

- https://github.com/luothomas/Recreating_Flappy_Bird

使用素材：

- 貼圖：https://www.spriters-resource.com/mobile/flappybird/sheet/59894/
- 音效：https://www.101soundboards.com/boards/10178-flappy-bird-sounds
- Singleton 介紹與應用：

  https://www.youtube.com/watch?v=CPKAgyp8cno&ab_channel=ResoCoder
- 相關筆記：

  https://kendevlog.wordpress.com/2018/08/14/unity%E5%AD%B8%E7%BF%92%E7%AD%86%E8%A8%98%EF%BC%9A%E5%A6%82%E4%BD%95%E5%AF%A6%E7%8F%BEsingleton/
- FindObjectOfType 效能問題：

  https://learn.microsoft.com/zh-tw/windows/mixed-reality/develop/unity/performance-recommendations-for-unity?tabs=openxr