Danny 老師的 iOS 開發指南 ～ 中階篇

Memory Leak

記憶體洩漏

Danny Shen

什麼是 Memory Leak？

- 記憶體洩漏指的是，程式動態分配的記憶體在不再需要時，沒有被及時釋放，導致系統可用記憶體逐漸減少，最終可能導致程式崩潰或系統性能下降，也就是有沒有好好的對記憶體管理做最佳化
- 在很多時候，Memory Leak 也不會影響程式運作，甚至說不會被發現。但即使不影響最終結果，但長久以往，就會讓你的 APP 無法發揮應有的效能
- 若不懂它的原理，發生問題時，你就不知道為什麼會發生，也不知道如何解決

典型案例

- 這是我親身經歷過的，一個典型的案例：
  - 一個 透過線上即時對話討論 文件的 APP，其中有一個 PDF Reader 的功能
  - QA 表示她每次使用翻個 20 幾頁就 會閃退，
  - 客戶表示他沒有發生過，他們使用的文件也就是 20～30 頁，直到有一天他開啟了一個 100 頁的文件，翻到 60 幾頁就閃退了
  - RD 表示在開發時，用 Simulator 翻了 100 頁，也沒有閃退，無法重現
  - 於是他們找我協助檢查，後來才發現是頁面顯示資料中的循環引用(交互參照)問題

為什麼有了 ARC 還是會有 Memory Leak 問題？

- Closure / NSTimer 捕獲
  - 當我們建立 Closure 或其他物件時，如果外部變數的生命週期比閉包長，閉包就會持有對外部變數的強引用，導致外部變數無法被釋放
- 循環引用 (Retain Cycles)
  - 當兩個或多個物件互相強引用，導致彼此的引用計數永遠不會降為零，通常是使用 weak 或 unowned 等弱引用來打破循環引用
- 其他
  - 如 CADisplayLink捕獲，第三方庫設計問題，橋接其他語言，而需要手動管理記憶體等

Memory Leak 物件變數補獲

以 Closure 與 Timer 為例

Danny Shen

物件變數補獲

- 我們在建立 Closure 或 Timer 的實體時，實際上會把 self 參照進去，若時生命周期有所交互，造成實際上循環引用 (Retain Cycles)
- 解決這個問題最簡單的方法，就是 self 使用 weak 參照
- 所謂的 weak 就是不使用 ARC 只有在母層存在時，才會有效，若母層沒有了，就馬上釋放的一種機制
- 要注意的是 Timer，回傳時，若上層的 self 不見了，請記得中斷 timer 本身，不然生命周期仍不會結束

Memory Leak 循環引用

Server 與 Client 的範例

Danny Shen

循環引用 (Retain Cycles)

- 循環引用 (Retain Cycles / 交互參照) 是最典型的造成 MemoryLeak 的原因，但在很多應用中，我們必需要循環引用，特別是母子類型的物件
- 例如有兩個 class 叫 Server 與 Client 他們在建立時，因為要找到對方，所以都是交互參照， Server 可以有多個 Client，但如果 Server 不見了，Client 就沒有任何存在的意義了，此時 Server 對 Client 就應該使用強參照，而 Client 對 Server 就應該用 weak 參照
- 若沒有依這個原則，就有可能會造成記憶體使用上的很大差異