CHAPTER 6-1

UIImage View：顯示圖像方塊

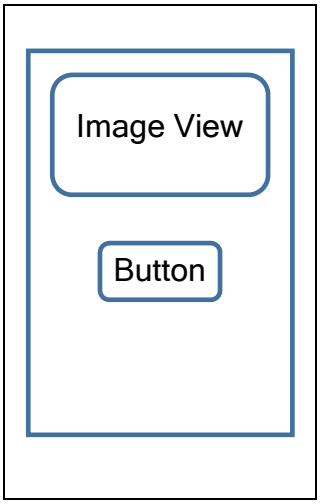
在加入圖像方塊後顯示圖片，點選按鈕後更換圖片

學習概念：

1. 首先用 IB 建立【圖像方塊】。
2. 將【操作按鈕】及【圖像方塊】與【程式碼】連結。
3. 最後在實作檔中相關程式，於處理載入後所觸發的事件，也就是撰寫利用【操作按鈕】結合【圖像方塊】，直接顯示在【設計畫面】上的程式。

Interface Builder

▼設計畫面



▼原始碼編輯器

```
@IBOutlet weak var myImageView:
UIImageView!
@IBAction func tapBtn(sender: AnyObject) {
```

▼設定顯示訊息視窗在文字標籤中

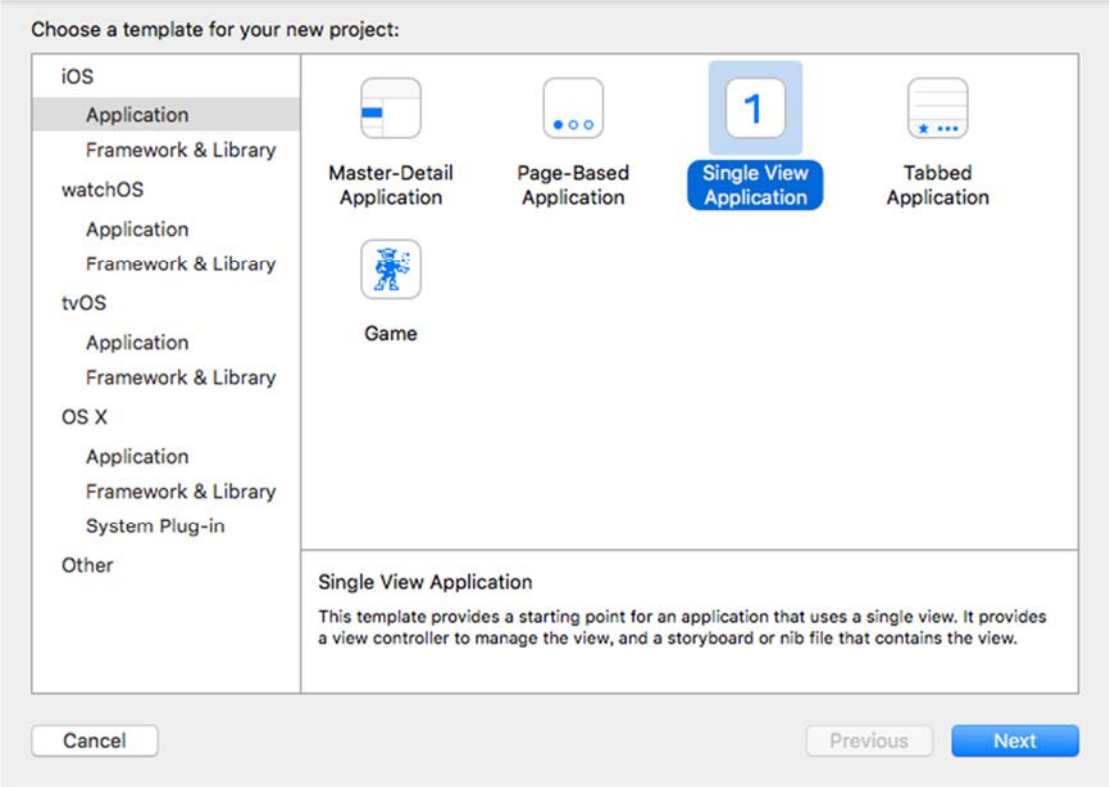
```
myImageView.image = UIImage(named:
"banana.png")
```

【執行結果】

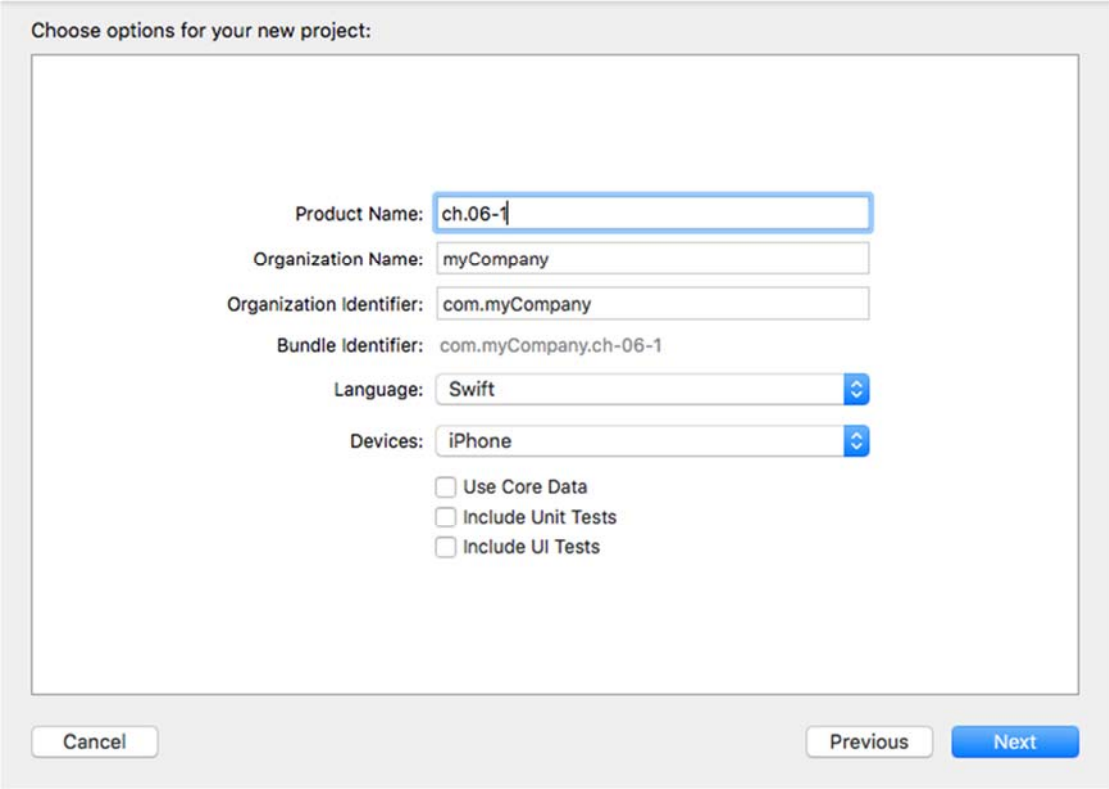
當 App 執行後，預先顯示 apple.png 圖片，點選【操作按鈕】後，讓【圖像方塊】顯示 banana.png 圖片在〈設計畫面〉中。

Step.1

開啟 xcode 時會出現的畫面，點選 iOS 下的【Application】，接著右視窗選擇【Single View Application】，點選【Next】選項後進入設定的基本視窗。

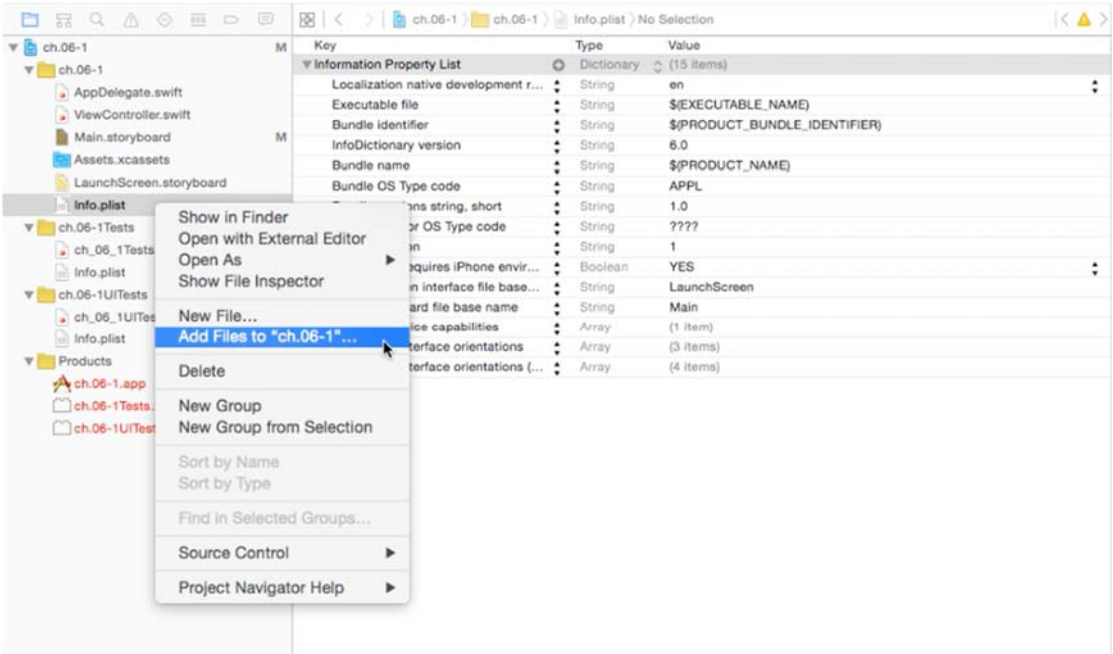


檔名及名稱設定，請將【Product Name】設定為 ch.06-1

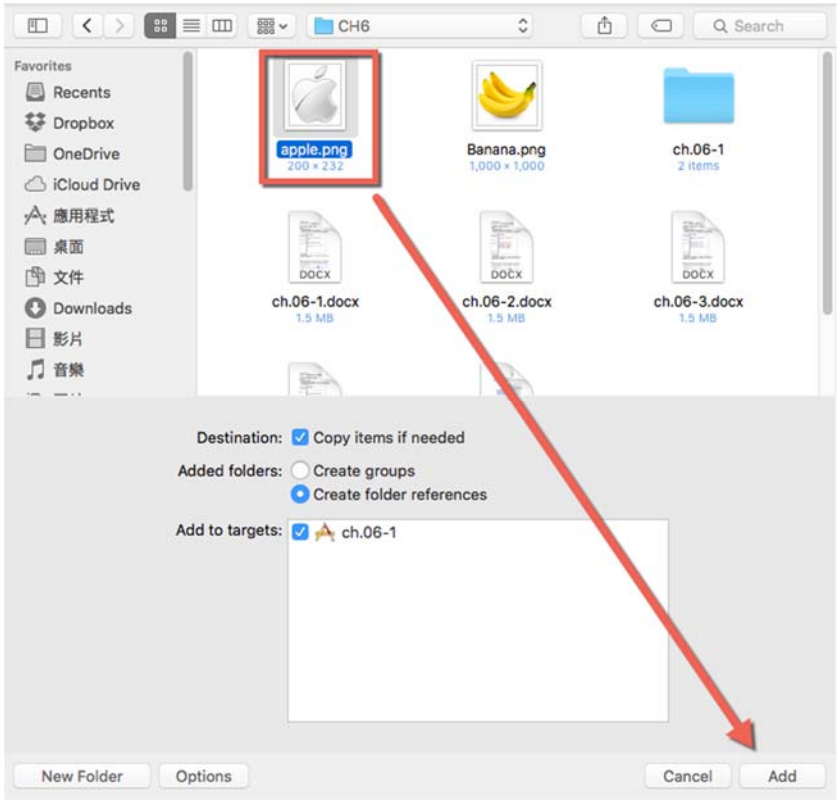


Step.2

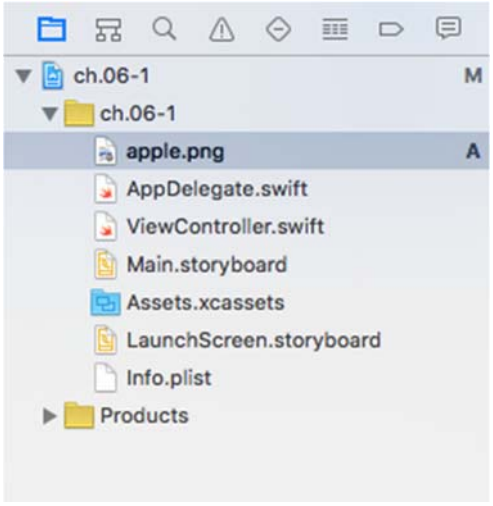
(1) 在這裡我們選擇一張圖片，展開 ch.06-1 目錄，在此目錄中任一位置按右鍵→Add Files to “ch.06-1….”。



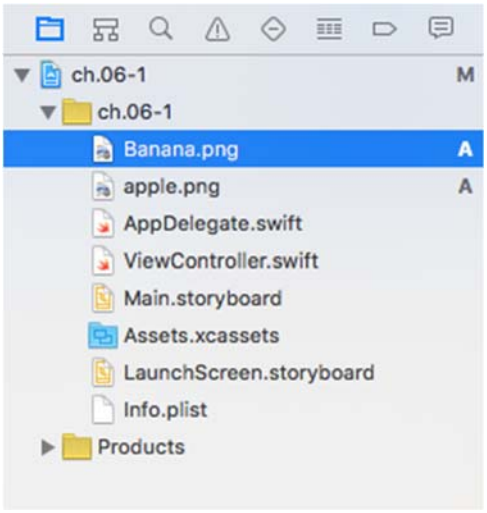
(2) 在這裡我們選擇選取的檔案名稱為【apple.png】選取後按「Add」。



(3) 完成後，接著會看到顯示出個檔名為＜apple.png＞。

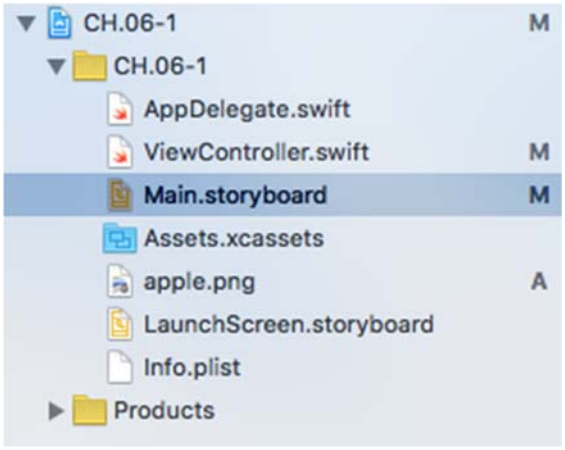


(4) 同樣的方法，將 Banana.png 也加入。

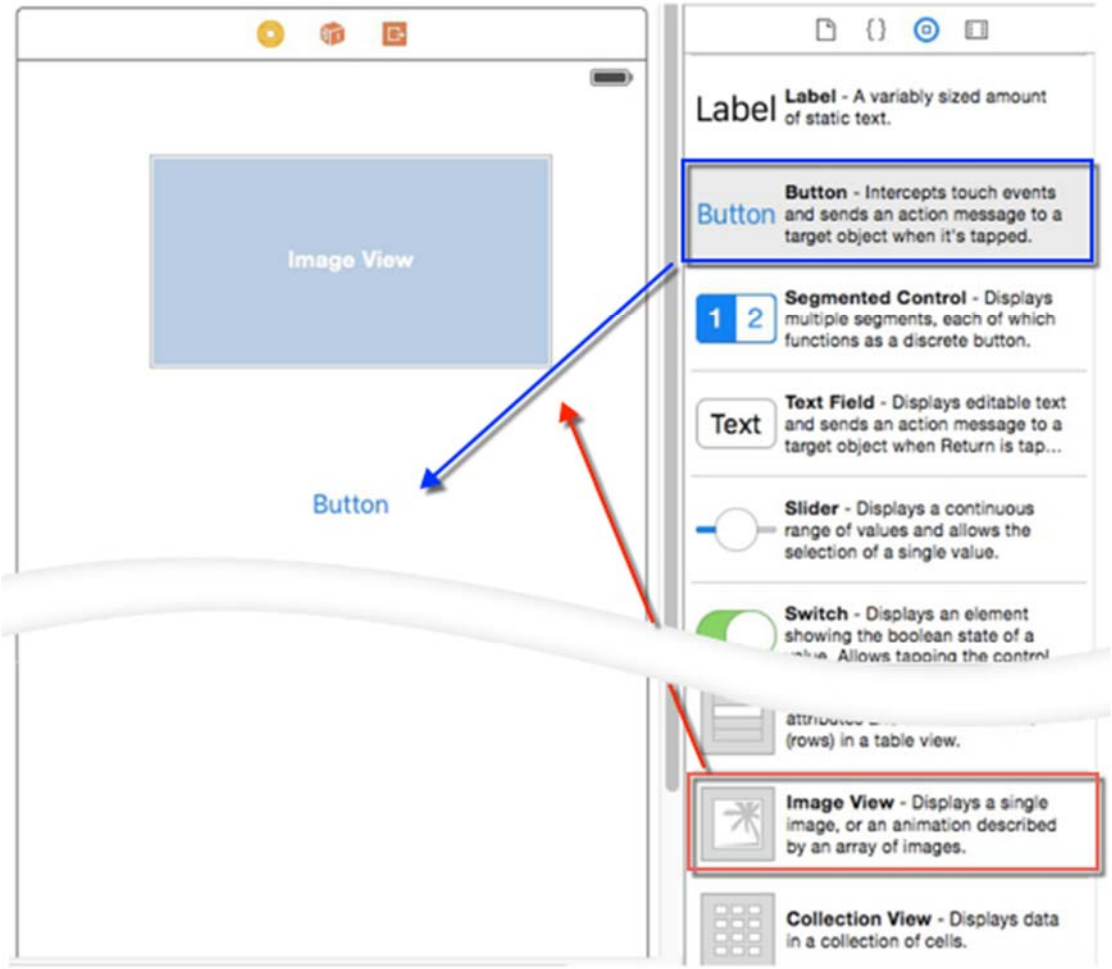


Step.3

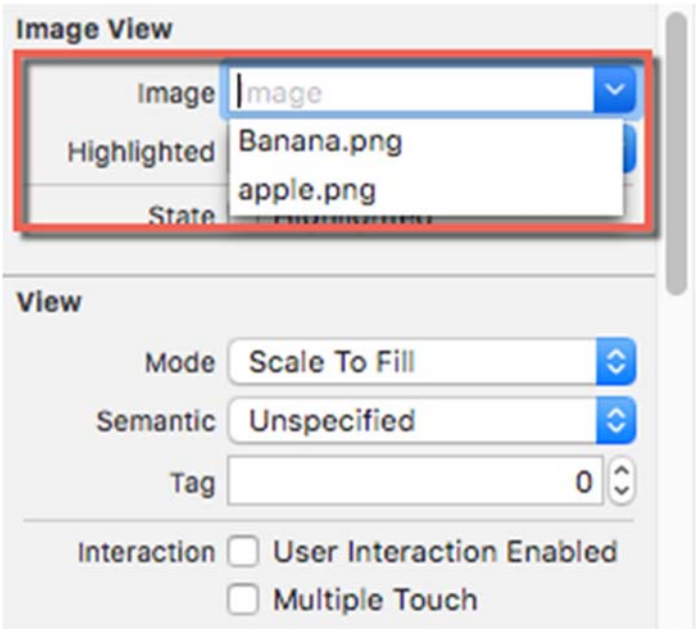
選取【Main.storboard】



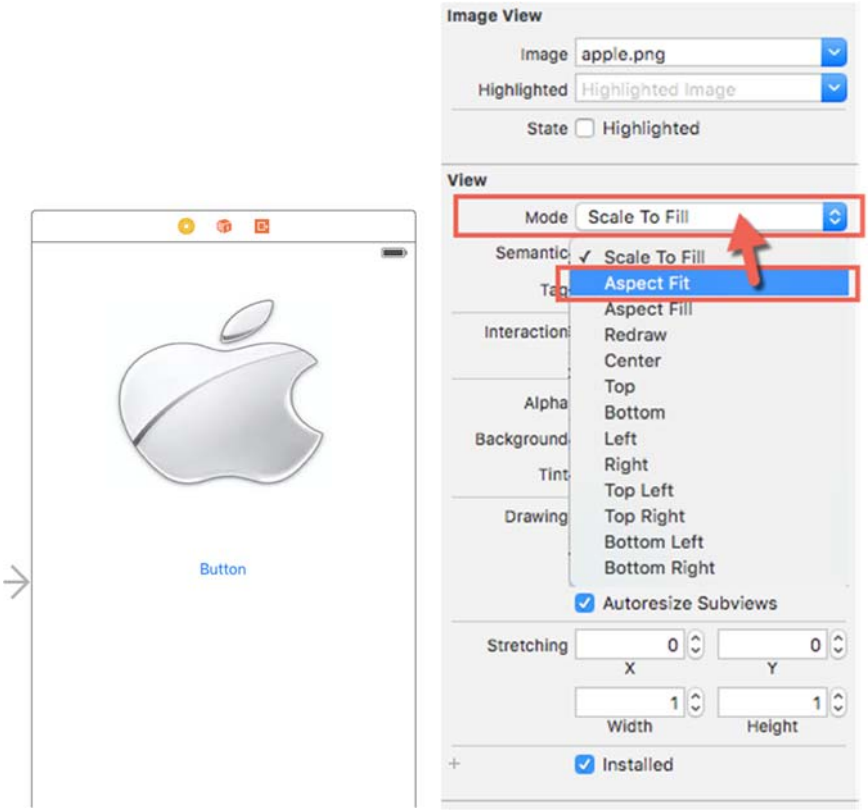
本章操作已選擇【iPhone 4.7-inch】 操作頁面。(詳見 5-1 屬性設定小技巧)
從【物件區】中拖曳【Button】及【Image View】到 IB 畫面中。



(1) 點選新加入的 ImageView，我們在右邊「屬性檢視」的視窗，點選我們所要加入點片的檔案名稱＜apple.png＞



(2)　在經由「屬性檢視」視窗加入後，隨即會發現在【設計畫面】的視窗會顯示我們所指定的＜apple.png＞圖像，此時此蘋果因填滿整個畫面而有變形現象，可以透過設定模式 mode 來選擇 Aspect Fit，保持原圖比例。



(3)　經過調整後顯示原先比例的圖形。

Step.4.

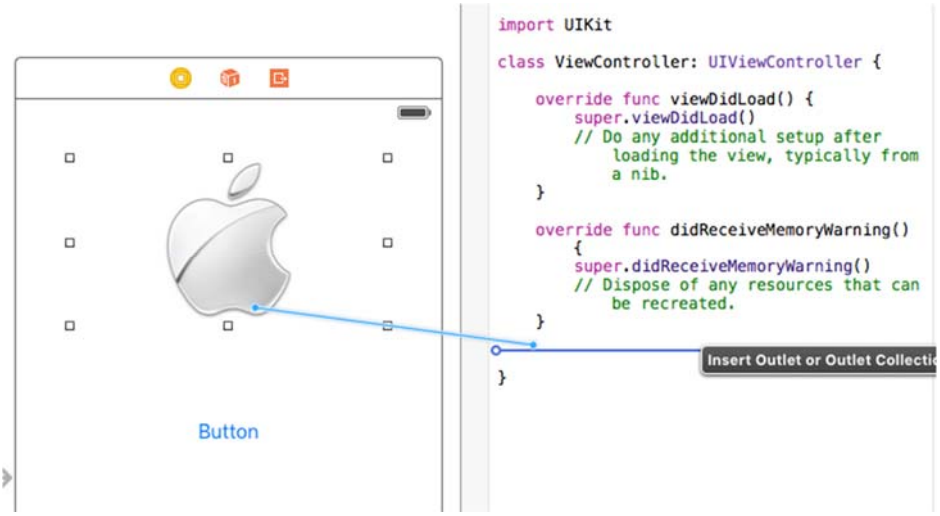
接著點選右上方工具列視窗【輔助編碼器】，就是雙圈符號【2】，進行程式碼編輯。

（詳見 5-1 屬性設定小技巧）

Step.5

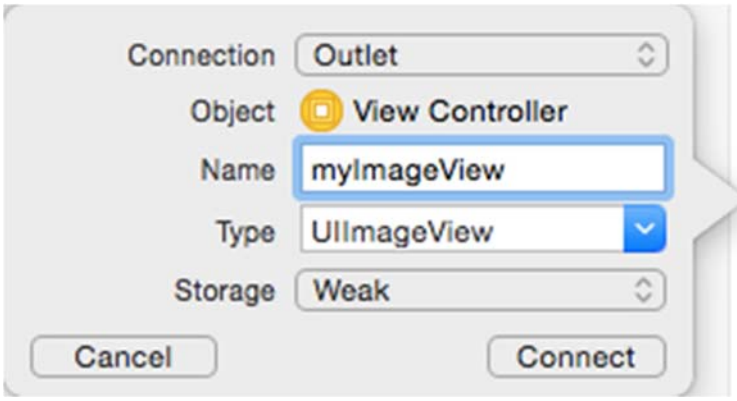
(1) 透過連結即產生的程式碼，控制 IB 建立的元件和【程式碼】連結。

(2) 將【圖像方塊】「Image View」與「變數名稱」連結。



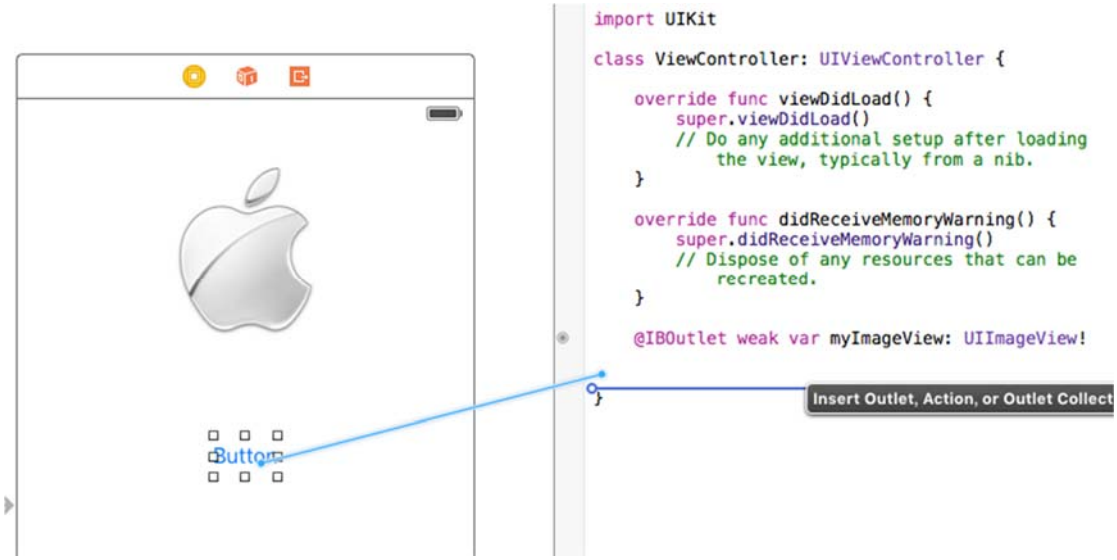
按住【Control】用滑鼠拖曳 Image View 元件。

在「Connection」欄位點選【Outlet】、「Name」欄位設定名稱【myImageView】以及在「Type」欄位點選【UIImageView】後，按【Connect】按鈕。



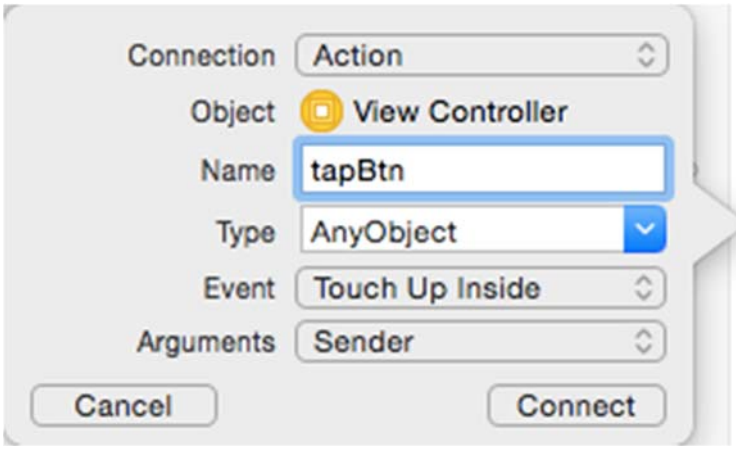
在拖曳後自動彈跳的視窗

(3) 將【操作按鈕】「Button」與「變數名稱」連結。



按住【Control】用滑鼠拖曳 Button 元件。

在「Connection」欄位點選【Action】、「Name」欄位設定名稱【tapBtn】以及在「Type」欄位點選【AnyObject】後，按【Connect】按鈕。



在拖曳後自動彈跳的視窗

(4) 將會自動插入程式碼，作為與 IB 的連結。

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading
            the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be
            recreated.
    }

    @IBOutlet weak var myImageView: UIImageView!

    @IBAction func tapBtn(sender: AnyObject) {

    }
}
```

連結後出現的程式碼。

Step.6

(1) 最後，我們在【紅框】中加入設定 swift 的程式碼，將透過串接執行程式後，將結果顯示在【圖像方塊】「Image View」中。

```
import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view,
            typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    @IBOutlet weak var myImageView: UIImageView!

    @IBAction func tapBtn(sender: AnyObject) {

        myImageView.image = UIImage(named: "Banana.png")

    }
}
```

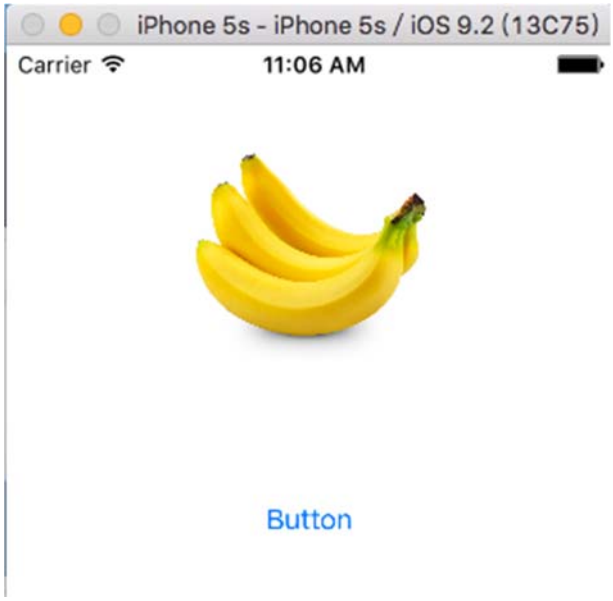
Step.7

在上方工具列按下【執行鍵▶】（Build and then run the current scheme），啟動模擬器執行程式。

Step.8

當 App 啟動後在顯示畫面時，在點選【操作按鈕】「Button」的同時，將會隱藏所設定的內容，隱藏後的結果會顯示在【圖像方塊】也就是「Image View」中的圖片內容。

試試按一下【操作按鈕】「Button」後會出現什麼樣的結果～

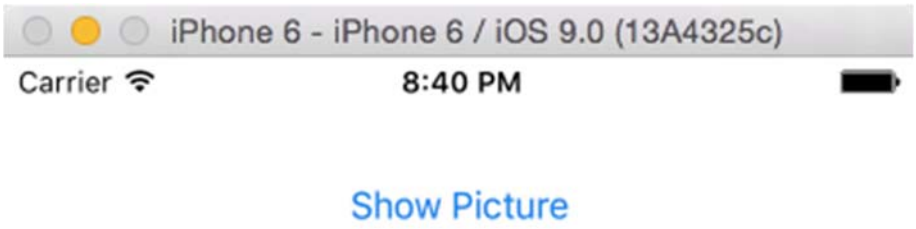


在經過我們點選【操作按鈕】「Button」時，則將原先蘋果圖像替換為香蕉圖像。

自 我 練 習

實作執行後結果：

這次呢，我們要練習的是運用【Button】及【Image View】與程式結合執行，透過【Button】指令選項讓【Image View】根據我們所撰寫的內容；在執行與【程式碼】連結並串接時，當執行【RUN】後，會「顯示」以及「隱藏」圖像的內容視窗。



在程式執行時，畫面中點選「Show Picture」後產生圖像。

然而，當在點選「Hidden Picture」後，則會隱藏出現顯示圖像畫面的結果。