

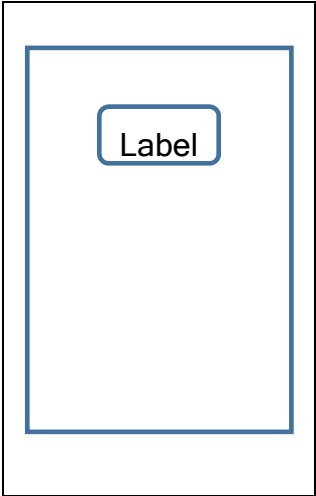
CHAPTER 6-4

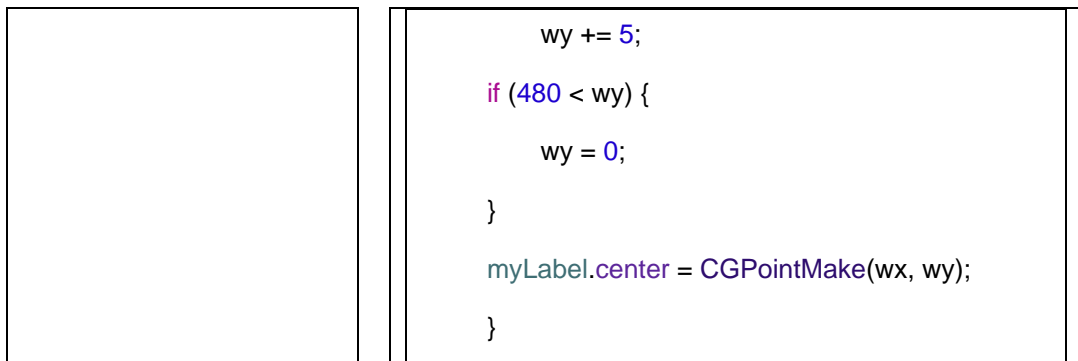
UIView：計時器動畫之文字標籤移動效果

利用計時器讓移動產生動畫

學習概念：

1. 首先用 IB 建立【文字標籤】。
2. 將【文字標籤】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【文字標籤】結合【程式碼】及撰寫程式，讓【文字標籤】產生移動效果，顯示在【設計畫面】上的程式。

Interface Builder	原始碼編輯器
<p>▼設計畫面</p> 	<p>▼設定文字標籤顯示在訊息視窗中移動</p> <pre>myTimer = NSTimer.scheduledTimerWithTimeInterval (0.05, target: self, selector: Selector("moveLabel"), userInfo: nil, repeats: true) func moveLabel() { var wx,wy:CGFloat; wx = myLabel.center.x; wy = myLabel.center.y; wx += 10; if (320 < wx) { wx = 0; } }</pre>

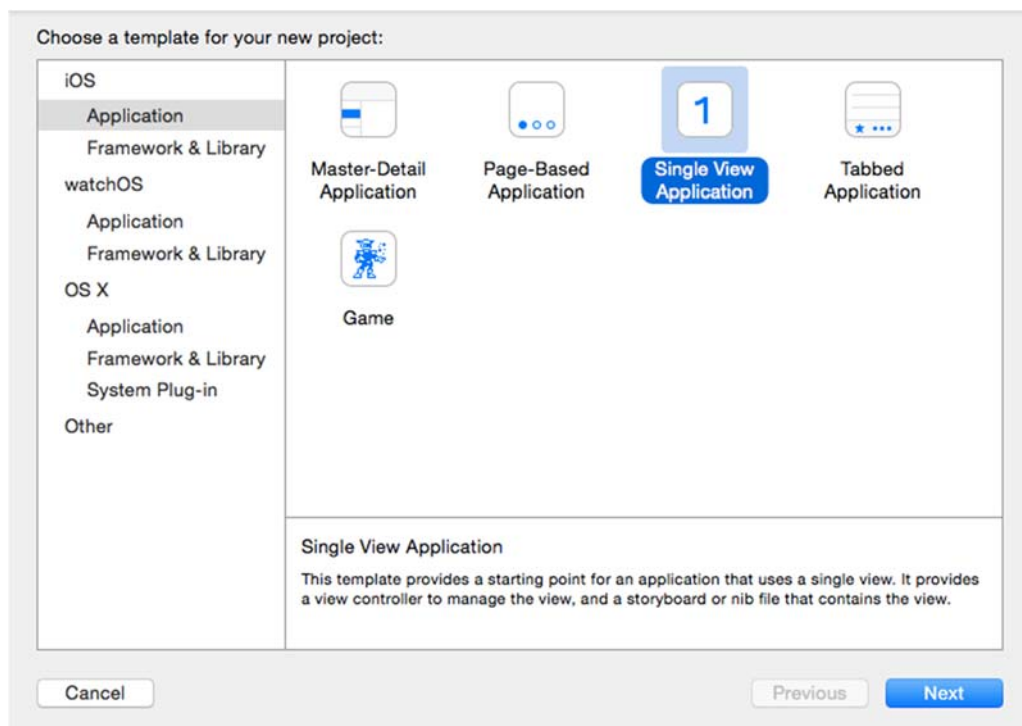


【執行結果】

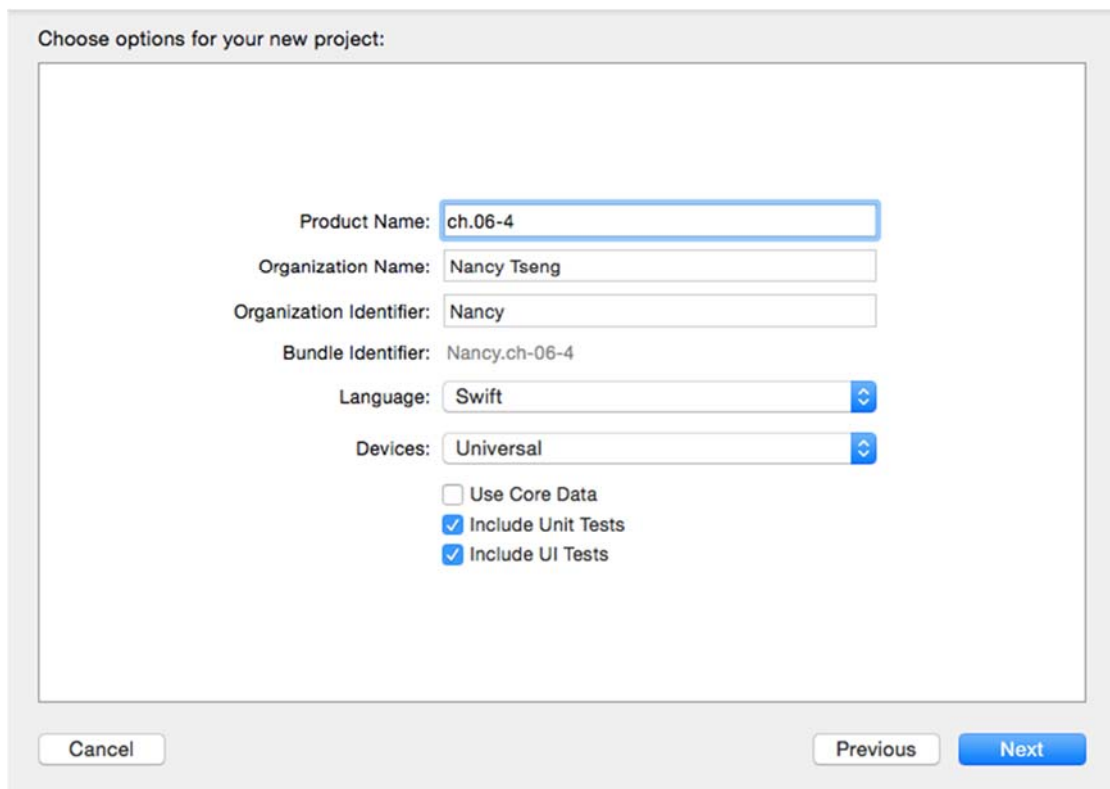
當 App 執行後，點選【文字標籤】後，讓【文字標籤】因為移動而產動畫的效果，顯示在〈設計畫面〉中。

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.06-4



The image shows the 'Choose options for your new project' dialog box in Xcode. The 'Product Name' field is highlighted with a blue border and contains the text 'ch.06-4'. Other fields include 'Organization Name' (Nancy Tseng), 'Organization Identifier' (Nancy), and 'Bundle Identifier' (Nancy.ch-06-4). The 'Language' dropdown is set to 'Swift' and the 'Devices' dropdown is set to 'Universal'. At the bottom, there are three checkboxes: 'Use Core Data' (unchecked), 'Include Unit Tests' (checked), and 'Include UI Tests' (checked). Navigation buttons 'Cancel', 'Previous', and 'Next' are at the bottom of the dialog.

Choose options for your new project:

Product Name: ch.06-4

Organization Name: Nancy Tseng

Organization Identifier: Nancy

Bundle Identifier: Nancy.ch-06-4

Language: Swift

Devices: Universal

☐ Use Core Data

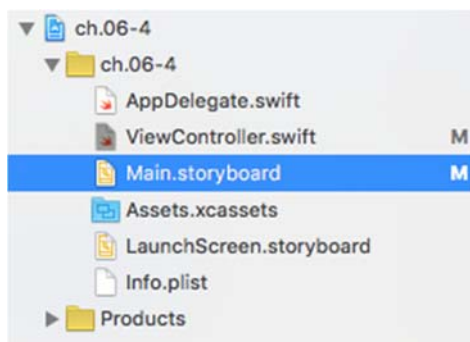
☒ Include Unit Tests

☒ Include UI Tests

Cancel Previous Next

Step.2

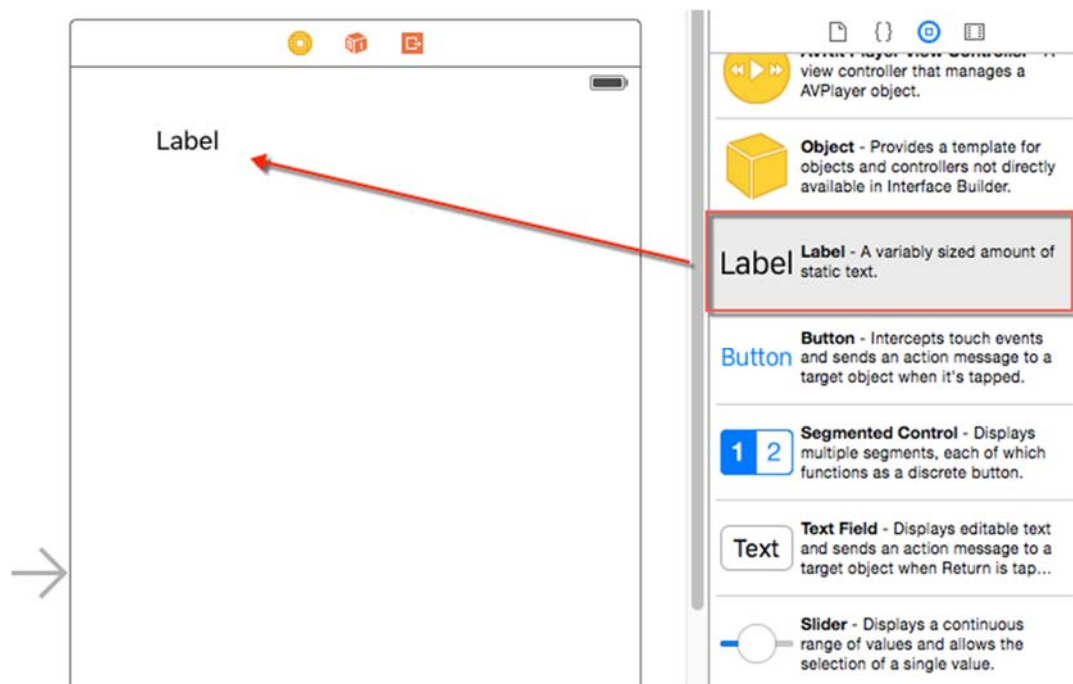
選取【Main.storyboard】



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)

Step.3

從【物件區】中拖曳【文字標籤】「Label」到 IB 畫面中。



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。

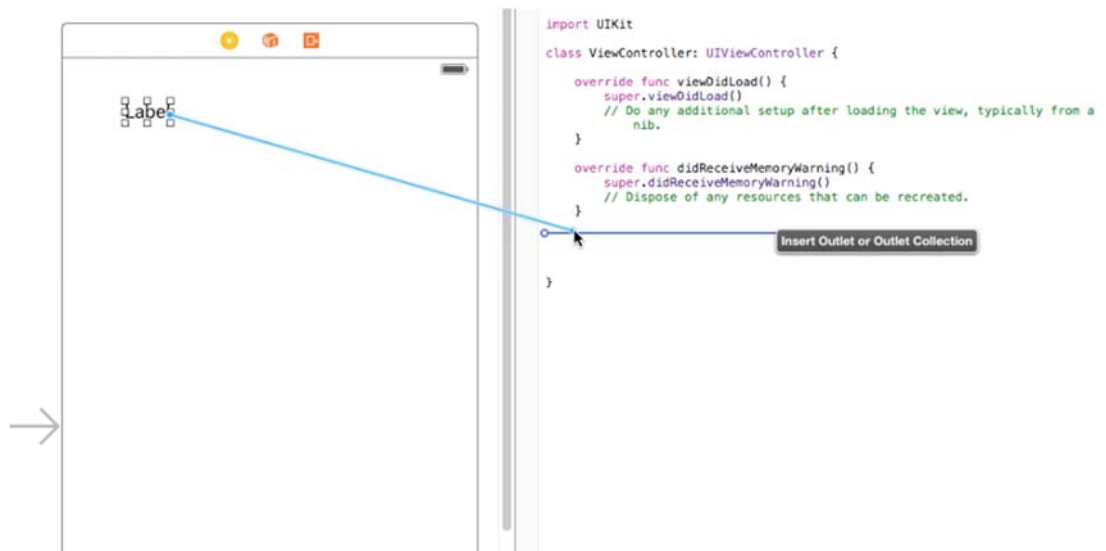


(詳見 5-1 屬性設定小技巧)

Step.5

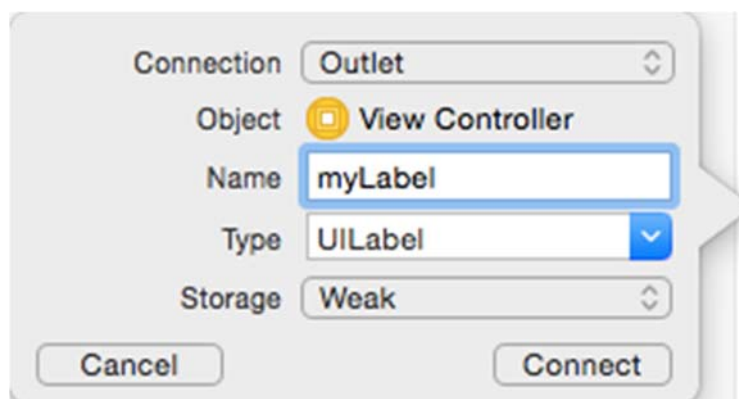
(1) 透過連結即產生的程式碼，控制 IB 建立的元件和【程式碼】連結。

(2) 將【文字標籤】「Label」與「變數名稱」連結。



按住【Control】用滑鼠拖曳 Label 元件。

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



在拖曳後自動彈跳的視窗

(3) 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 myLabel: UILabel!
}
```

連結後出現的程式碼。

Step.6

(1) 接下來，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，讓結果顯示在【文字標籤】「Label」中。

// 宣告 myTimer 為 NSTimer 型別的計時器

```
var myTimer : NSTimer = NSTimer()
```

// 定義 myTimer 實際的內容

```
myTimer =
```

```
NSTimer.scheduledTimerWithTimeInterval
```

```
(ti: NSTimeInterval,           //執行的秒數
```

```
target: AnyObject,             //作用對象
```

```
selector: Selector,            //執行的 function
```

```
userInfo: AnyObject?,          //使用者資訊
```

```
repeats: Bool)                  //是否要重複執行
```

```

import UIKit

var myTimer : NSTimer = NSTimer()

class ViewController: UIViewController {

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

        myTimer = NSTimer.scheduledTimerWithTimeInterval(0.05, target: self,
            selector: ("moveLabel"), userInfo: nil, repeats: true)

    }

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

    @IBOutlet weak var myLabel: UILabel!

    func moveLabel() {
        var wx,wy:CGFloat;
        wx = myLabel.center.x;
        wy = myLabel.center.y;
        wx += 10;
        if (320 < wx) {
            wx = 0;
        }
        wy += 5;
        if (480 < wy) {
            wy = 0;
        }
        myLabel.center = CGPointMake(wx, wy);
    }

}

```

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，畫面內容在出現【文字標籤】「Label」的同時，將會使動畫設定顯示於畫面中的內容結果。

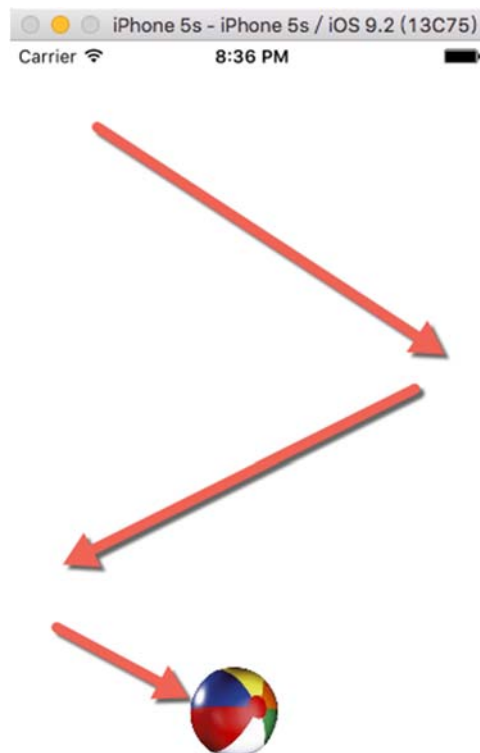


在執行「RUN」後，會出現「Label」移動時，
並且重覆的自左上向右下移動的顯木示效果。
（小編在這裡設定為 iPhone6 畫面，供各位讀者們參考）

自我練習

實作執行後結果：

接下呢，我們要練習的是運用【**UIImageView**】放入一張球的圖片與程式結合執行，透過程式讓【**UIImageView**】根據我們所撰寫的內容；點選【**RUN**】觸發與【**程式碼**】時，會啟動計時器所撰寫的動畫執行的內容視窗。



在啟動「**RUN**」後，會出現「球」不停的移動時，當遇到邊框時會自動彈回的效果。