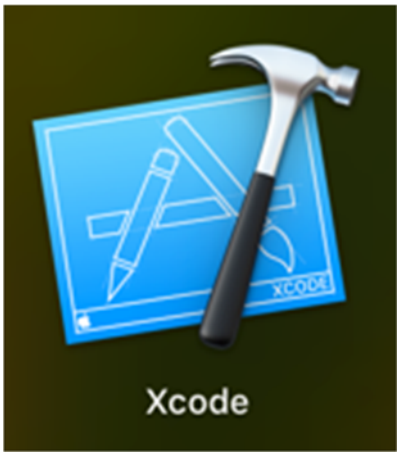
CHAPTER 3

3-1 專案的建立

建立專案

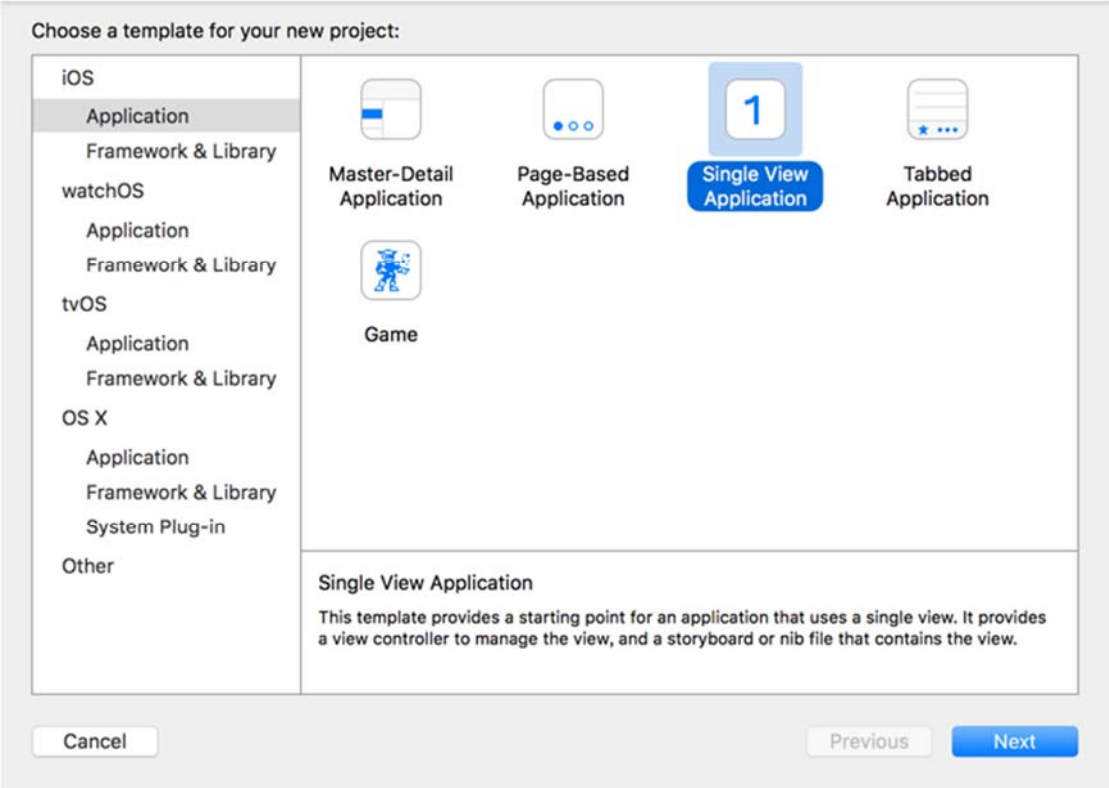
點擊應用程式中的 Xcode 後，即可帶出起使畫面。



第一個選項是 Get Started with a playground，主要是透過程式的模擬器可以輔助了解 swift 語法， 這部分在第四章時會介紹。第二個選項是 Create a new Xcode project，若要建立專案則由此進行。

右半部分是顯示最近開啟過的專案。

選擇專案樣本
在選擇了 Create a new Xcode project 後，會顯示五種系統內建的樣版，本書會用到前四種樣版，第五種樣版主要作為 Game 的開發，則不在本書中敘述。開發者可以選擇與自己 APP 最類似的樣本進行使用。



當選定好 Single View Application 後，即可點選 Next 進行下一步。

接下來需要針對此 APP 專案進行一些設定。
Product Name: 設定專案的名稱
Organization Name: 設定公司名稱， 若有申請 Apple Developer Account 則可以填入公司名稱
Organization Identifier: 設定組織的識別碼， 填入的通常就是公司的網址

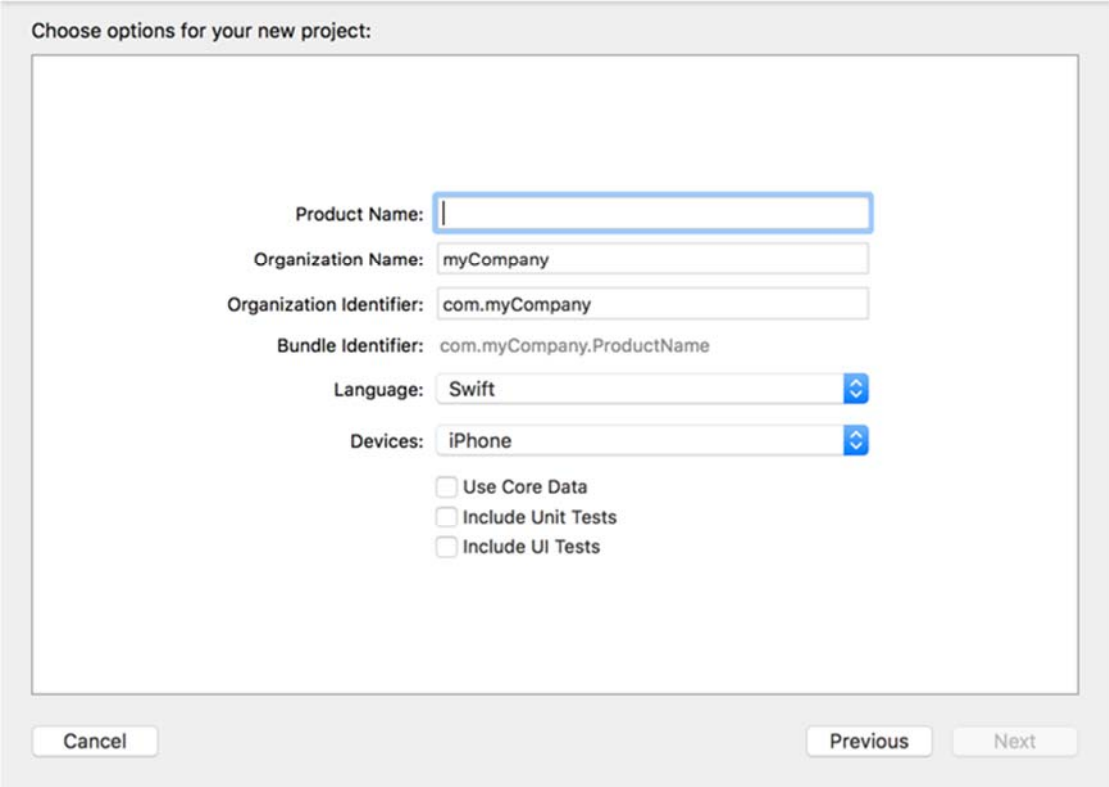
Bundle Identifier: 用來標示應用程式的唯一 ID，此部分由上方填入的 Product

Name 與 Organization Identifier 自動產生，通常是以反向的 DNS 方式命名的，

例如：com.myCompany.myApp

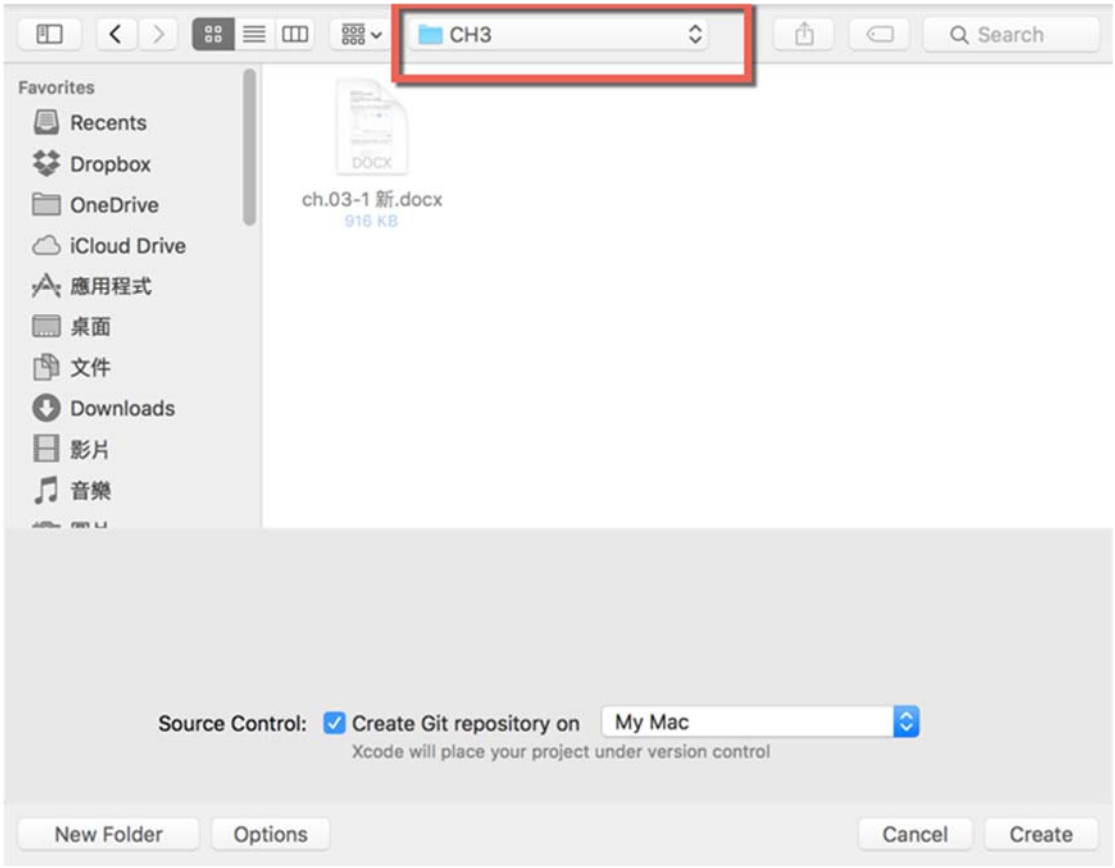
Language: 開發的語言，可以選擇使用 Objective-C 或 Swift
Device: 此 APP 適用的裝置，包含 iPhone, ipad 以及 Universal (同時適用於 iPhone 與 iPad

在下方有三個選項，則都不需要打勾
Use Core Data 作為若要將資料儲存於此 APP 時使用。
Include Unit Tests 專案中包含單元測試的元件
Include UI Tests 專案中包含使用者介面測試的元件

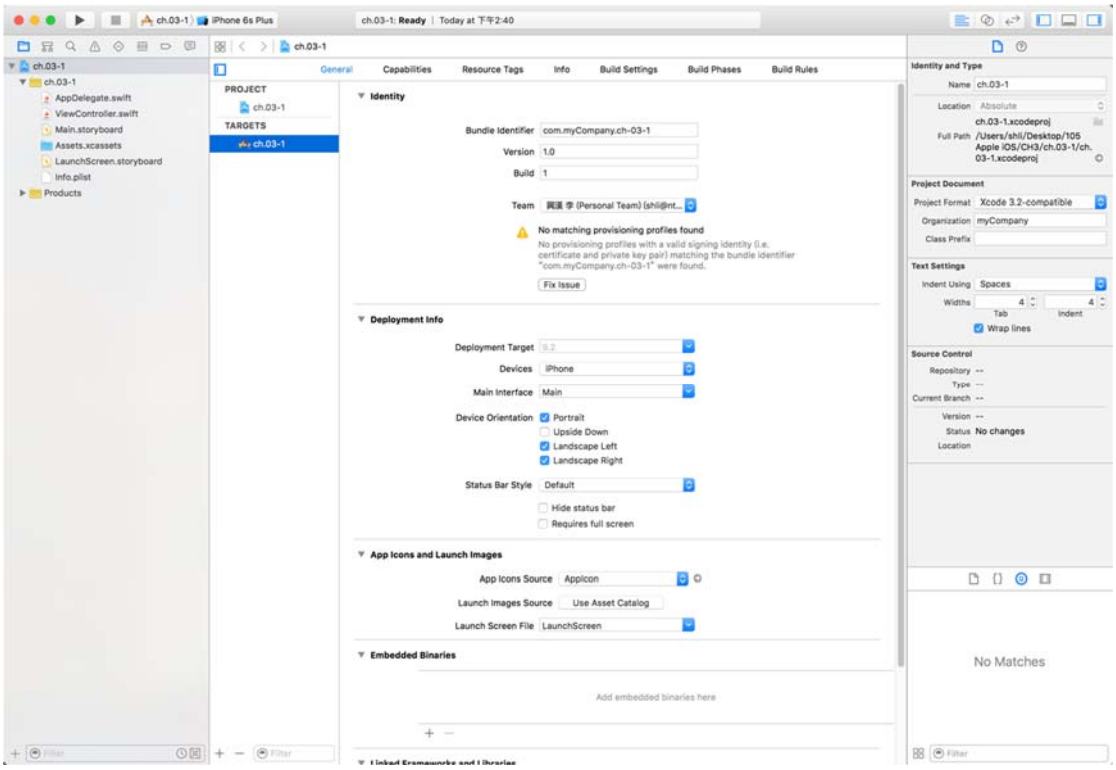


當輸入完檔案名稱後， Next 會顯現，則可以點擊進入下一步。

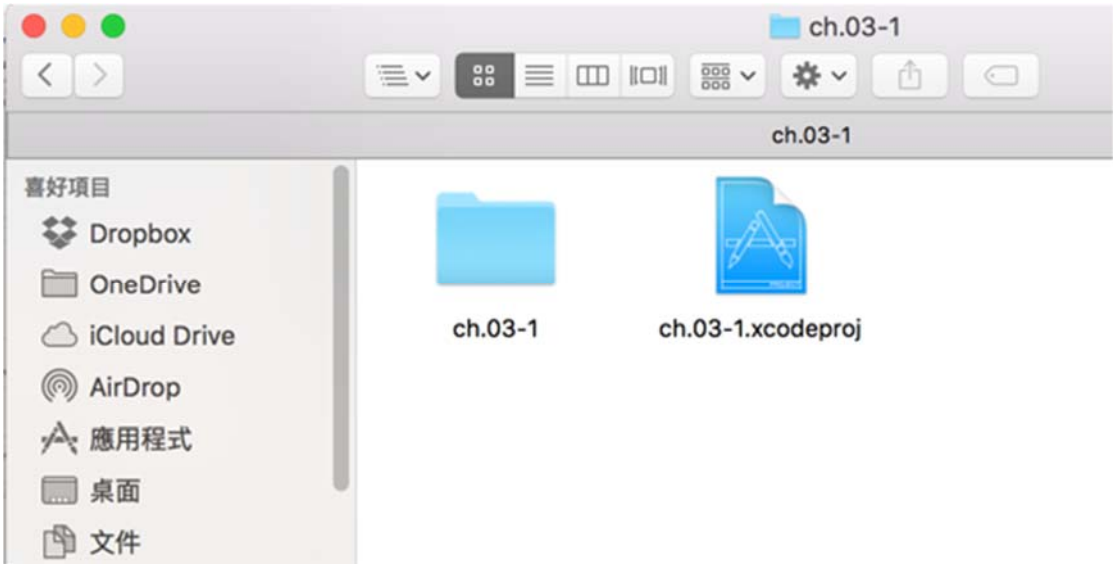
選擇專案要儲存的位置，選擇到適當的專案存放位置後，按下 Create



建立專案後，會帶出本專案的一般選項。

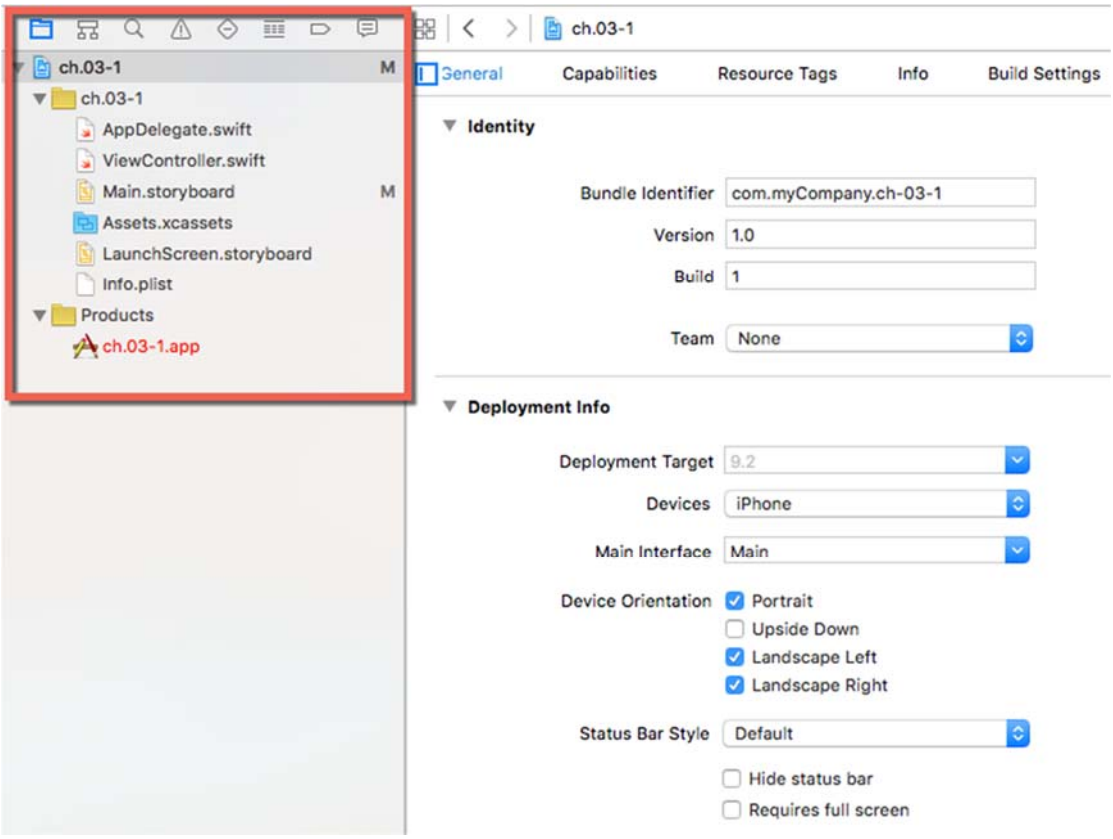


此專案也會建立於所選擇的目錄中
每個專案會有一個資料匣，以專案名稱命名，此資料匣內 ch.03-1.xcodeproj 是這個專案主要的檔案，另外 ch.03-1 則包含了專案所要用到的設定、函式庫、或一些使用者加入的影音圖檔等。



3-2 專案的結構

在 3-1 節中已經建立好專案，本單元將介紹專案的結構。



Main.storyboard

用來設計 APP 畫面的檔案，目前是使用單一設計畫面，在本書後續章節中將會介紹一個 storyboard 中包含多個 APP 設計畫面

ViewController.swift

開發者最重要的程式碼編寫所在處，可以用來撰寫 APP 設計畫面中各個元件的動作。

AppDelegate.swift

處理非使用者在 APP 畫面上的操作所產生的事件。例如：第一次執行 APP 時的動作；當電話進來或鎖屏時的操作；應用程式進入後台的動作；應用程式重新回到前台；應用程式要退出時的動作。讀者可將其視為 ViewController.swift 的上一層。

Assests.xcassets

可將圖片檔拖放置此處，另外向是製作 APP icon 等也都會存在於此處。

LaunchScreen.storyboard

作為 APP 啟動的畫面設計，在本書後續章節中提到製作啟動畫面時會用到。

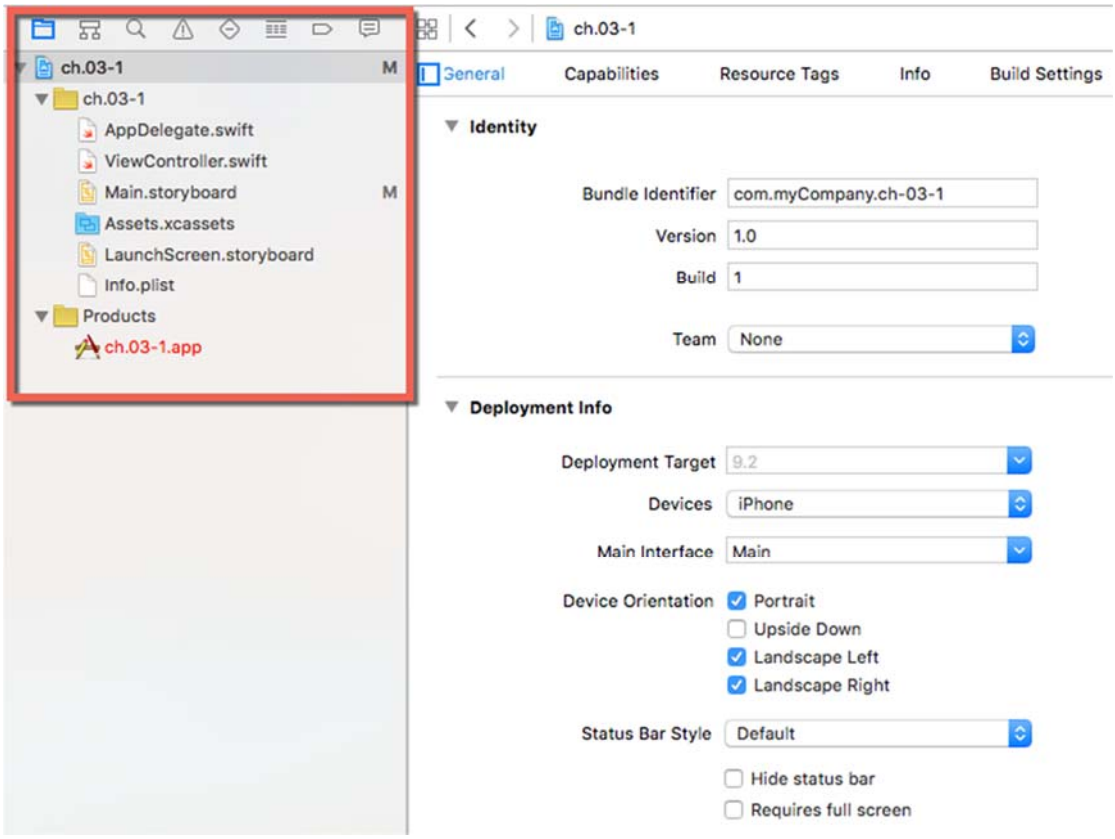
Info.plist

儲存此 APP 的一些參數設定。

Products

內存放專案名的 APP 執行檔，當開發者執行了 Build and then run the current scheme，系統會自動建立此檔案。

中，將先介紹固定手機尺寸的畫面配置。在第七章時會在介紹如何適用於多種

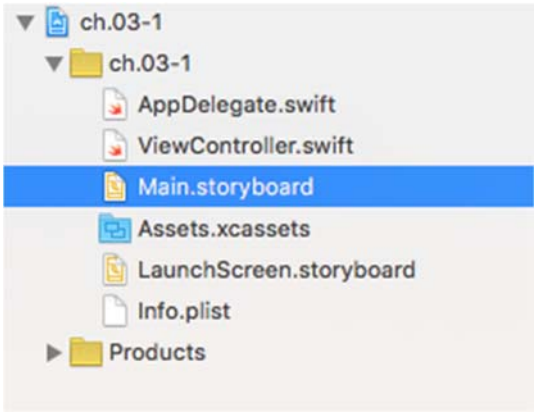


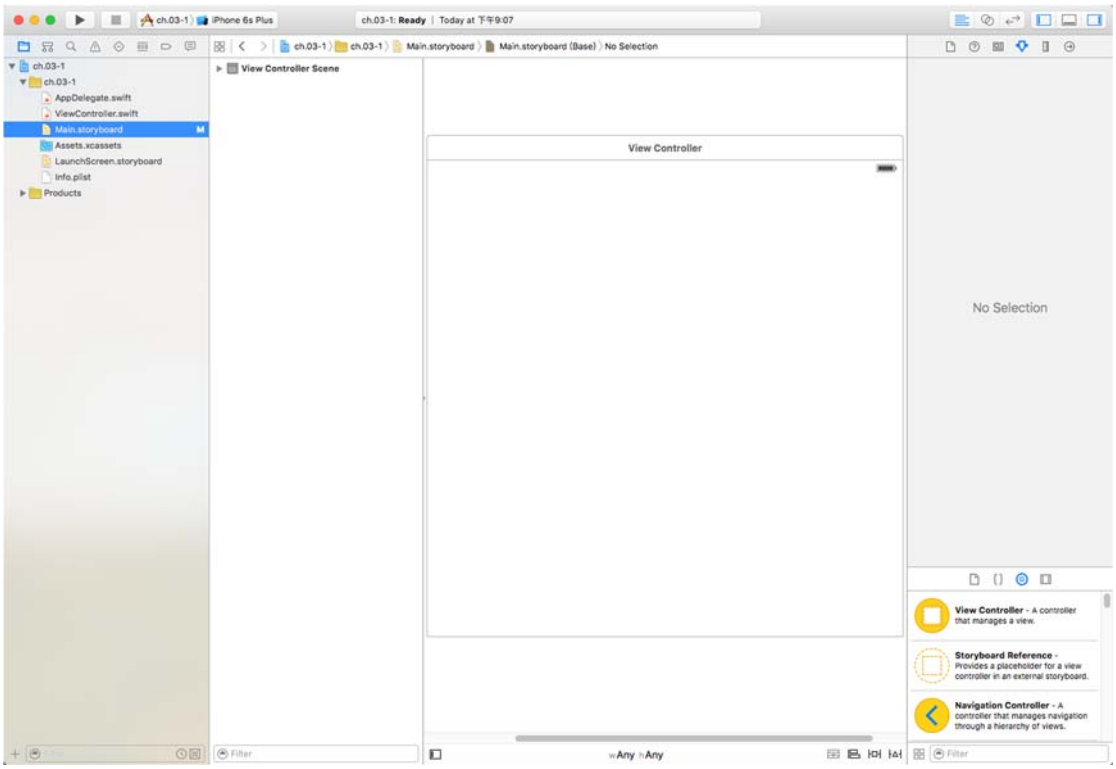
3-3 專案的開發配置

在 3-1 節中已經建立好專案，本單元將安排開發環境的配置。
為了教學上的需要，方便讀者能夠統一使用固定的編輯方式，因此在本章節中，將先介紹固定手機尺寸的畫面配置。在第七章時會在介紹如何適用於多種機型的畫面配置。

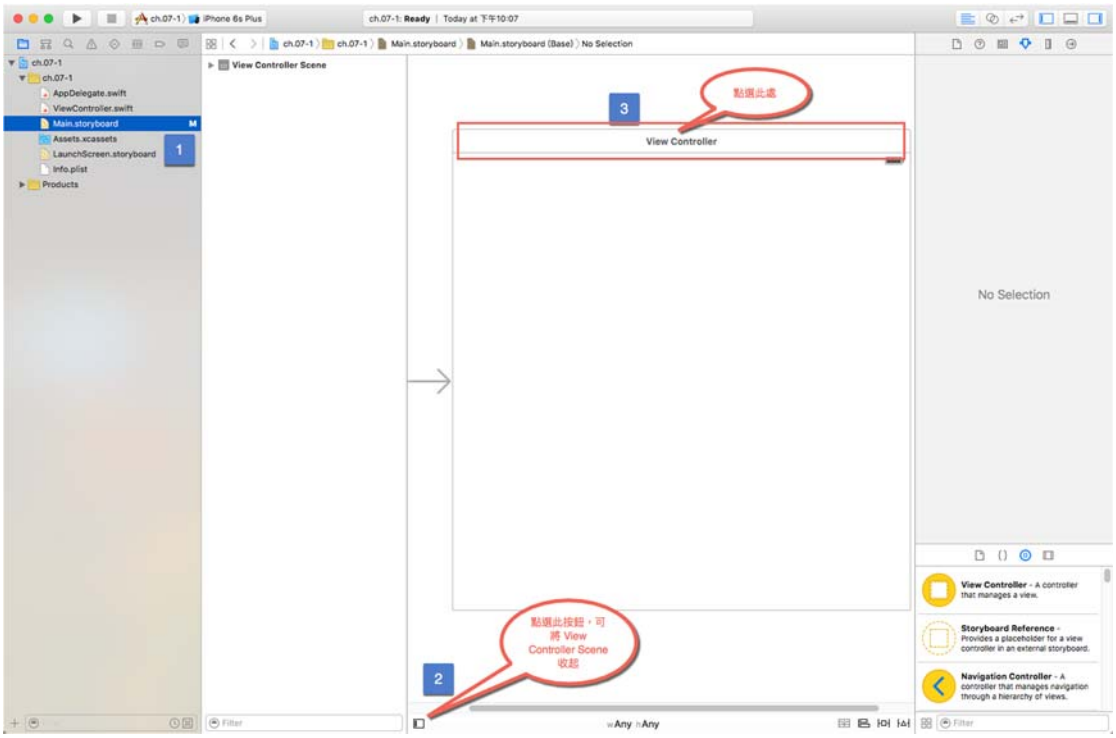
Step.1

在新建的專案中，由左上方選取【Main.storyboard】，螢幕中心將顯示空白的 APP 設計畫面，



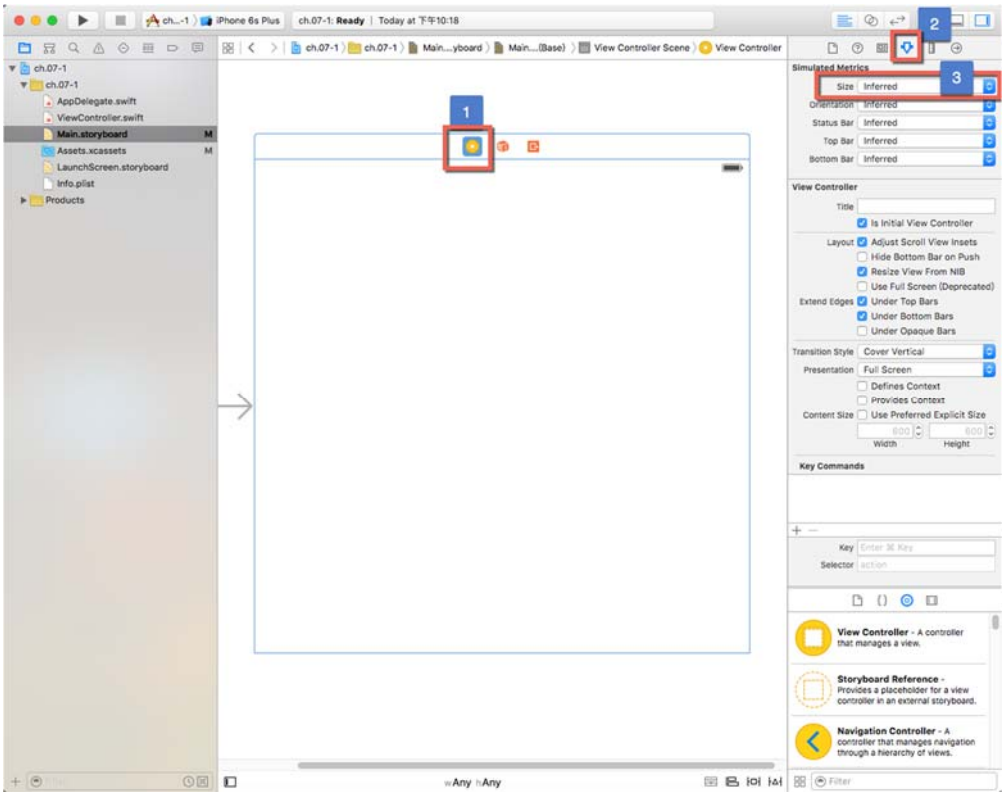


此時可以先將 View Controller Scene 先縮小，只先留下 APP 設計畫面。
點選此 APP 設計畫面的上半部 View Controller



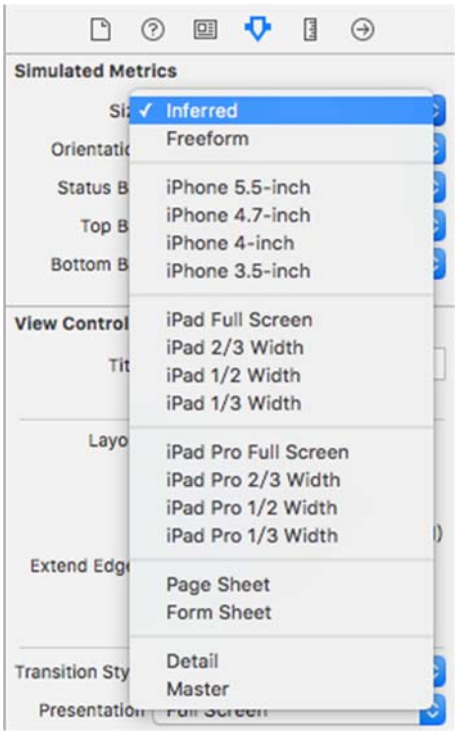
Step.2

選取【View Controller】，左半部會顯示，點選第四項 Show the Attributes Inspector，在第一項 Simulated Metrics 可以設定所要設計的畫面尺寸



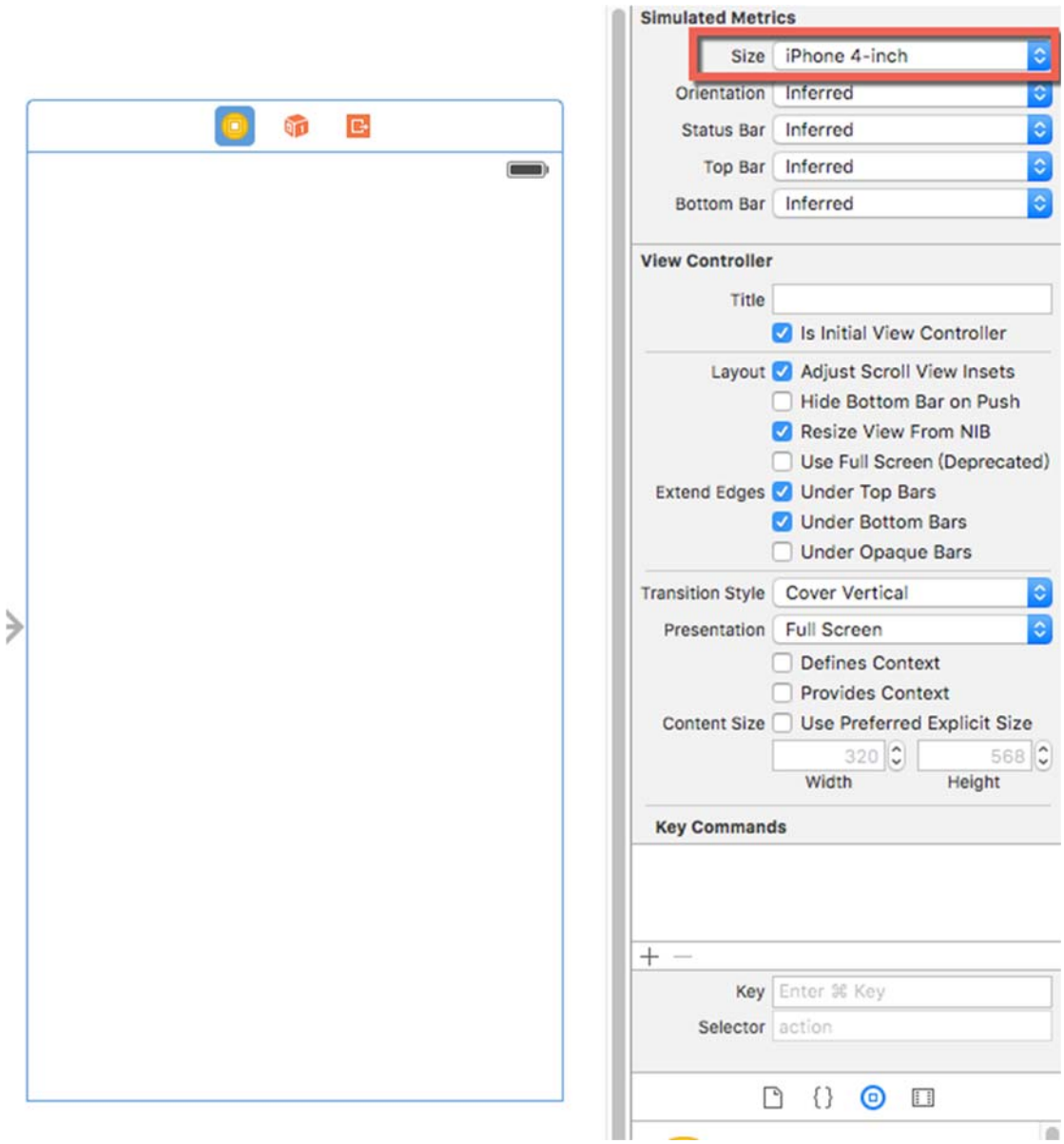
Step.3

在 Size 選項部分，可以看到 Size 選項就包含了 iPhone 與 iPad 的選項。
在 iPhone 中的選項是依照型號而決定尺寸，在 iPad 部分則是依照所佔用螢幕的比例，如 1/3 Width, 1/2 Width, 2/3 Width, full screen



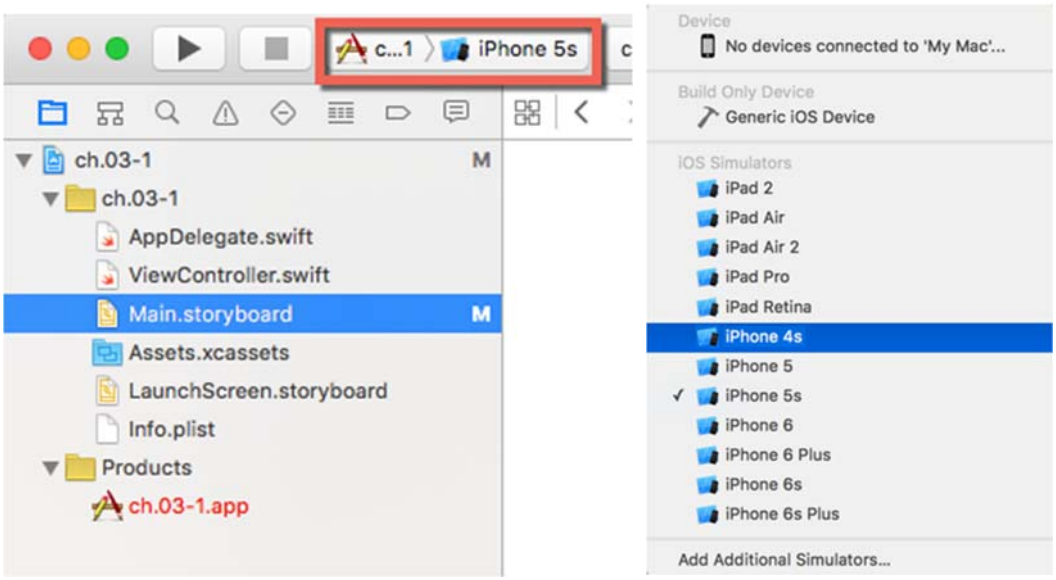
Step.4

選定 iPhone 4-inch，則可以看到中間的手機畫面就縮小為 4-inch 大小。



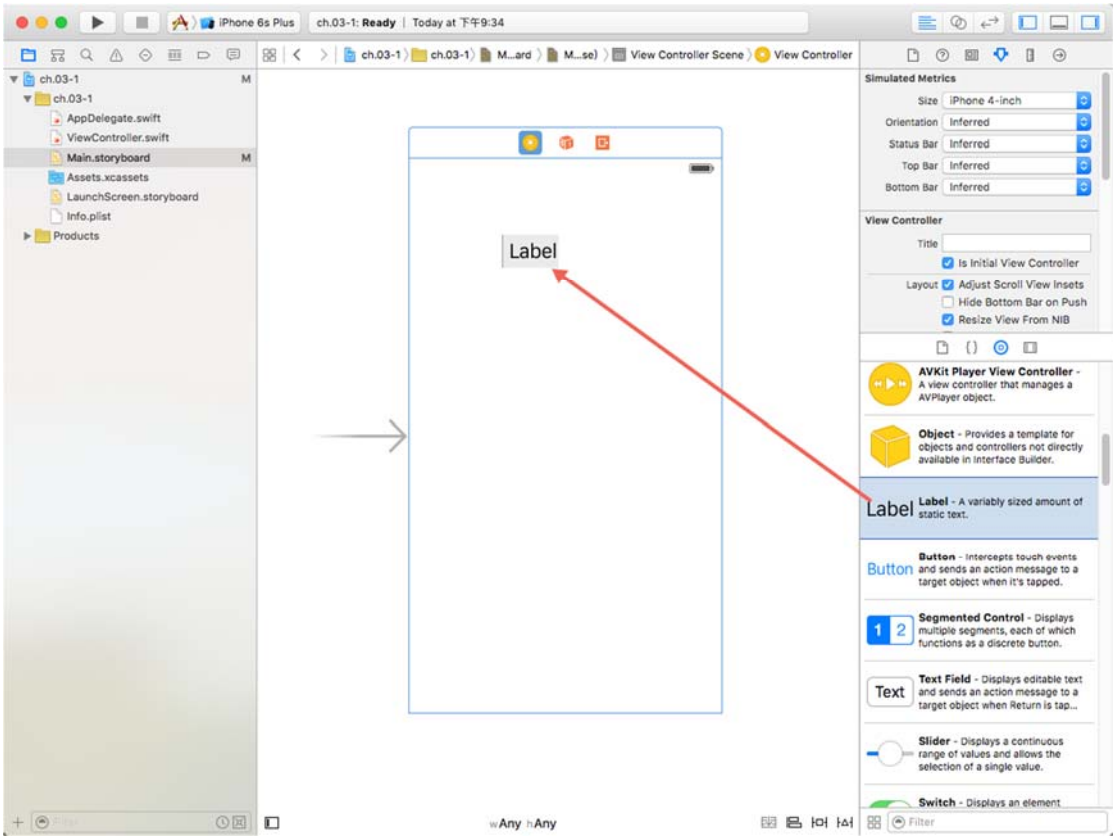
Step.5

在 XCode 左上方也需要選定模擬機的型號，建議依照所選定的編輯畫面尺寸來選擇適當的模擬機型號，這樣才不致於畫面變形。



Step.6

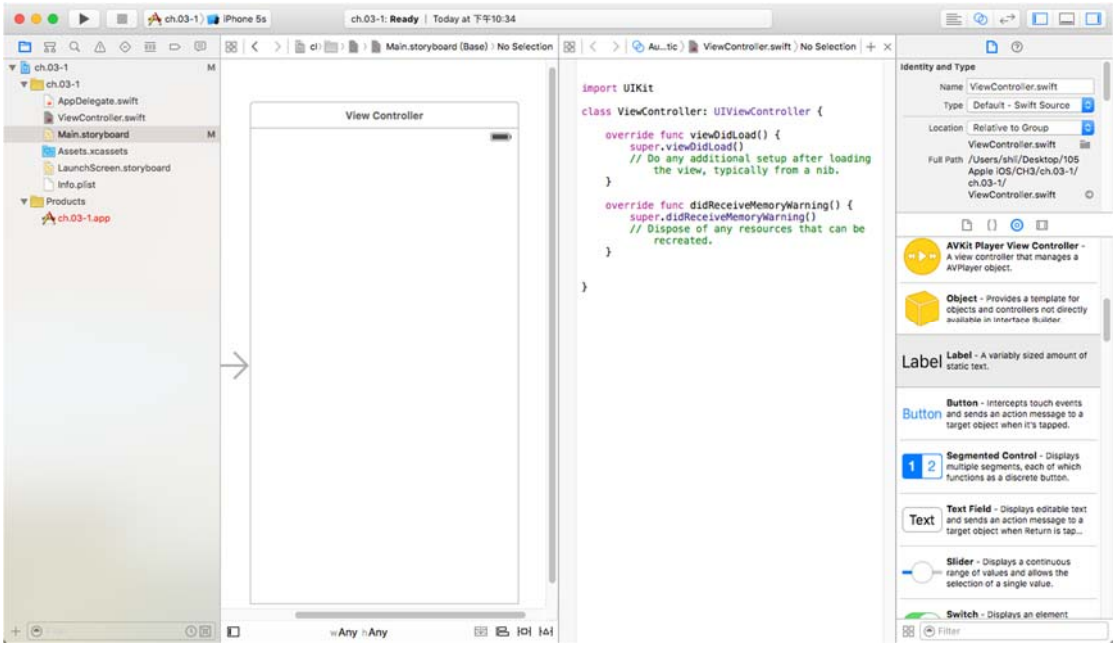
此時的 APP 設計版面配置，則適合我們拖放右方的元件到 APP 設計版面



Step.7

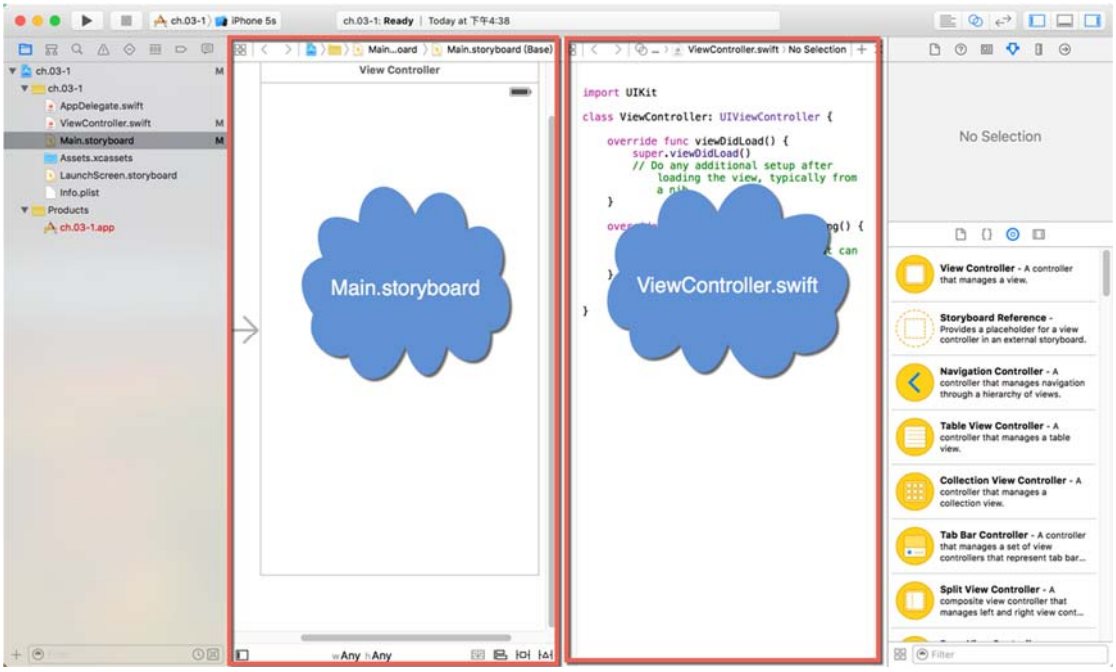
當 APP 設計版面配置了元件之後，則要進行程式的撰寫，這時建議讀者將中間視窗分割為二，此動作可以透過由上方的 show the assistant editor 切割畫面

建議讀者在每個新專案一建立之後，就先將設計畫面調整為如下之介面。



3-4 畫面與程式的連結

畫面與程式的連結主要是使用 Main.storyboard 與 ViewController.swift ，這也就是為何在 3-3 節最後，建議讀者在開發系統時能夠先將開發環境安排為



Interface Builder ▼設計畫面 Main.storyboard	▼原始碼編輯器　ViewController.swift
畫面上配置的元件	定義資料項目（含變數名稱）和動作名稱
	動作的程式碼

畫面上的元件與程式碼的連結

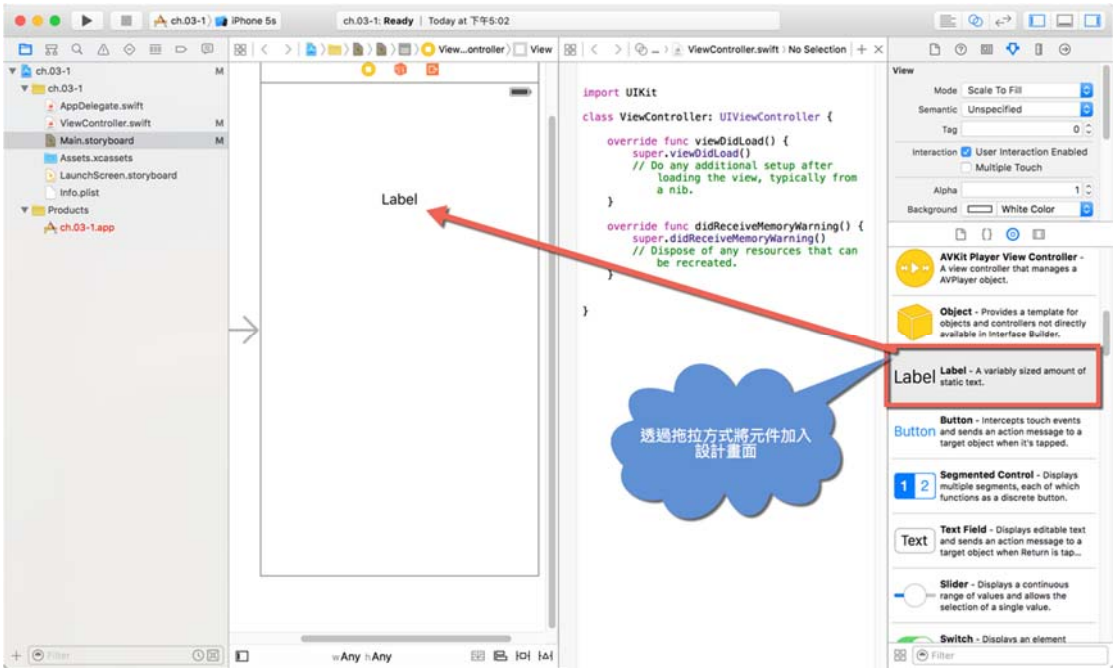
首先，使用 Interface Builder 以在畫面上配置使用者介面的元件，接下來我們就需要將這些元件與程式碼連結起來，以便透過程式來控制這些元件。

元件的操控分為兩種，
IBOutlet:只需要給予元件名稱，透過此名稱來呼叫此元件，以用作於設定此元件的屬性。
IBAction:需要給定此元件要執行的事件並定義名稱，並且需要撰寫程式碼已完成當觸發某事件後要做的事情。

IBOutlet 的連結

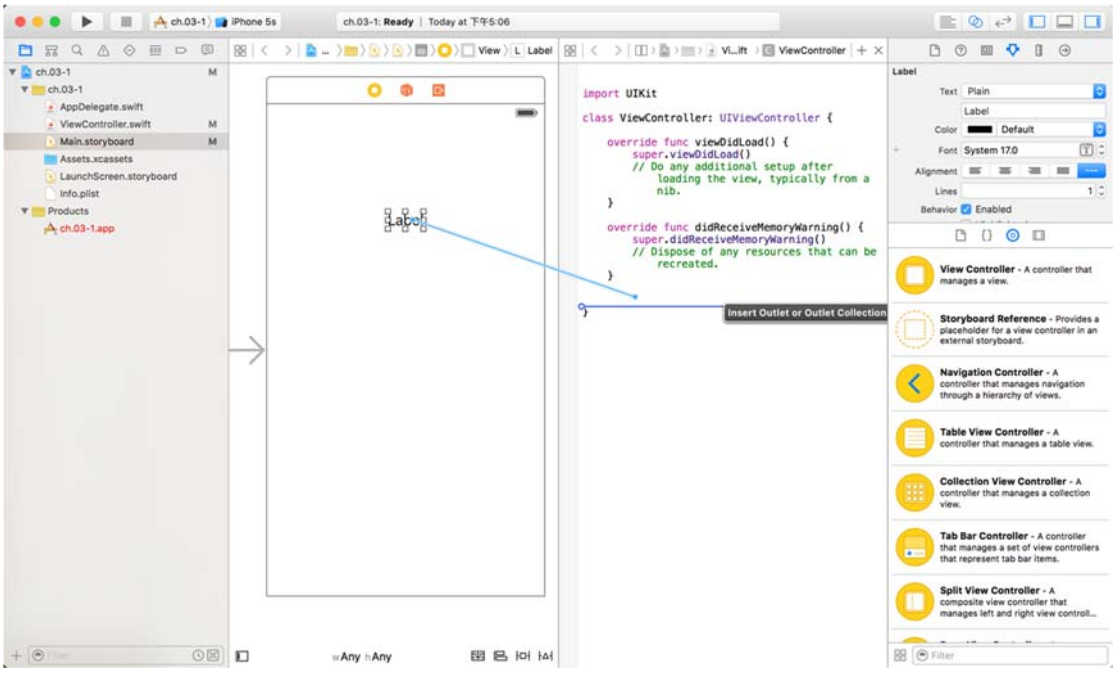
透過 IBOutlet 可以宣告元件名稱並建立連結, Xcode 會自動在 ViewController.swift 中插入相關的程式碼。

Step1. 加入一個 Label 元件到 設計畫面中。



Step2. 滑鼠游標移至此元件上，按下滑鼠左鍵放開，即可點選此新增加的物件，再按住『Control』鍵，再次按下滑屬左鍵後，點選不放，拖拉滑鼠至右方的程式設計區，可以看到產生出一條藍色的線條，將滑鼠移至程式區的下方『』』之前 （也就是最外層 {}之內） 鬆開滑鼠。

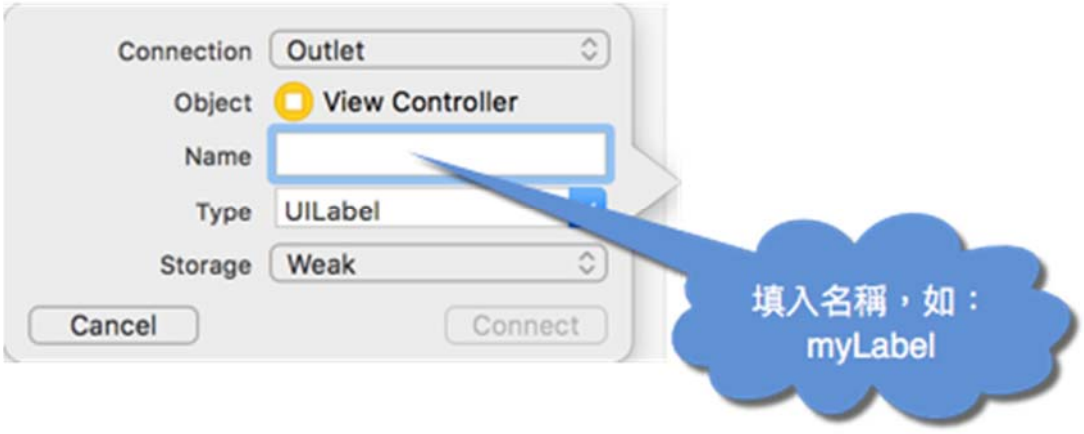
另一方法是在此元件上點選滑鼠右鍵按住不放，在拖拉滑鼠至左方程式設計區。



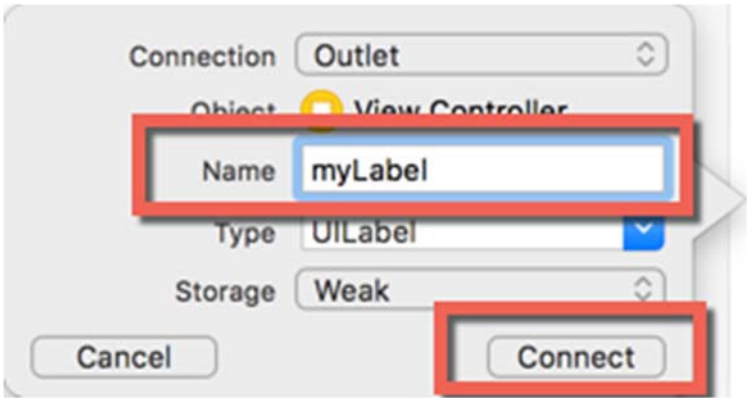
Step3. 鬆開滑鼠後，會產生一個對話視窗，此時可以於 Name 中填入此元件的名稱。

在本書中盡量 Outlet 的命名以名詞 my 作為名稱開頭。

例如：Label 元件， 則命名 myLabel； button 元件，則命名 myBtn。



Step4. 當填入 myLabel 於此元件的名稱後，按下 Connect。



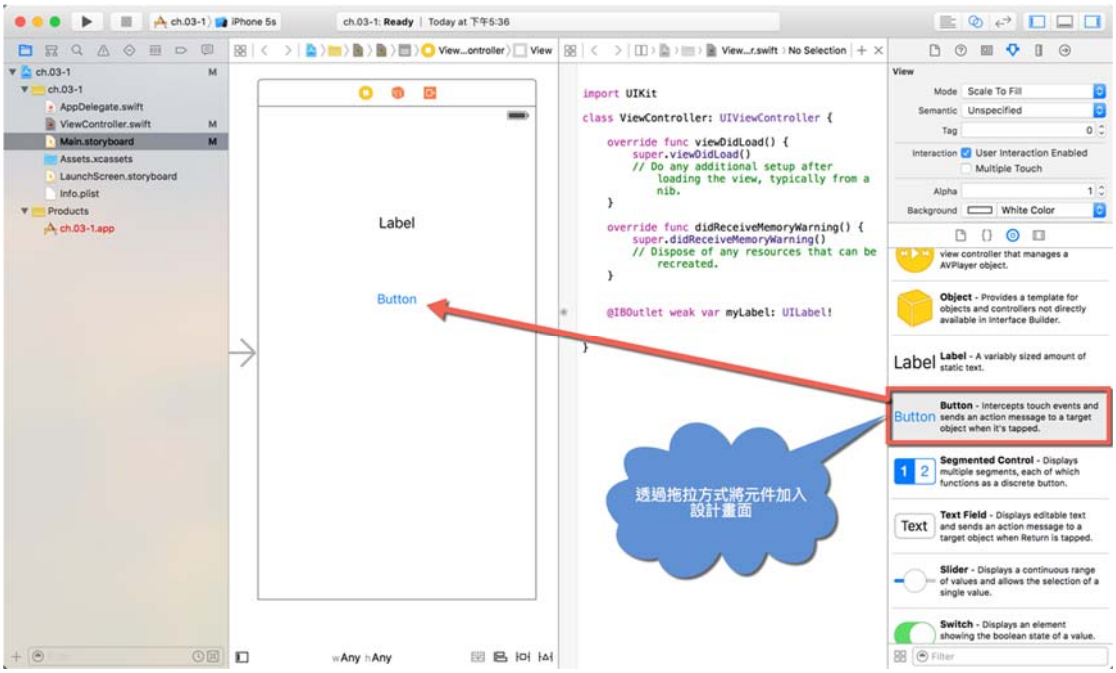
Step5. 在右方的程式設計區，將會自動插入一段程式碼。 此即完成了此 Label 的命名。

```
@IBOutlet weak var myLabel: UILabel!
```

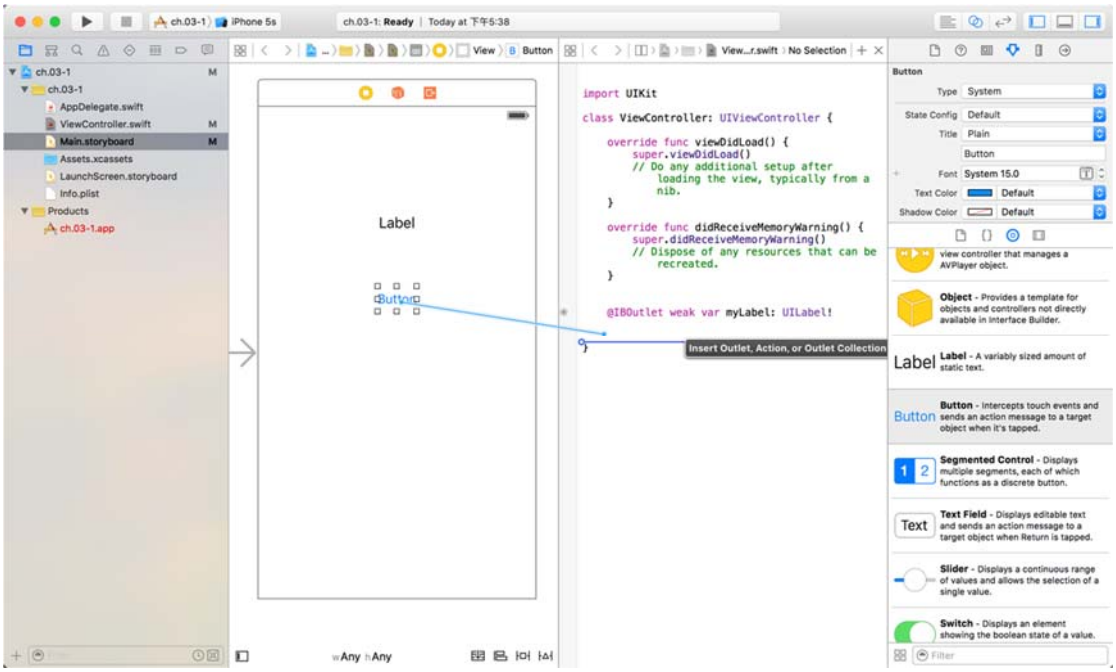
IBAction 的連結

透過 IBAction 可以宣告元件的方法名稱並建立連結, Xcode 會自動在 ViewController.swift 中插入相關的程式碼。

Step1. 加入一個 Button 元件到 設計畫面中。

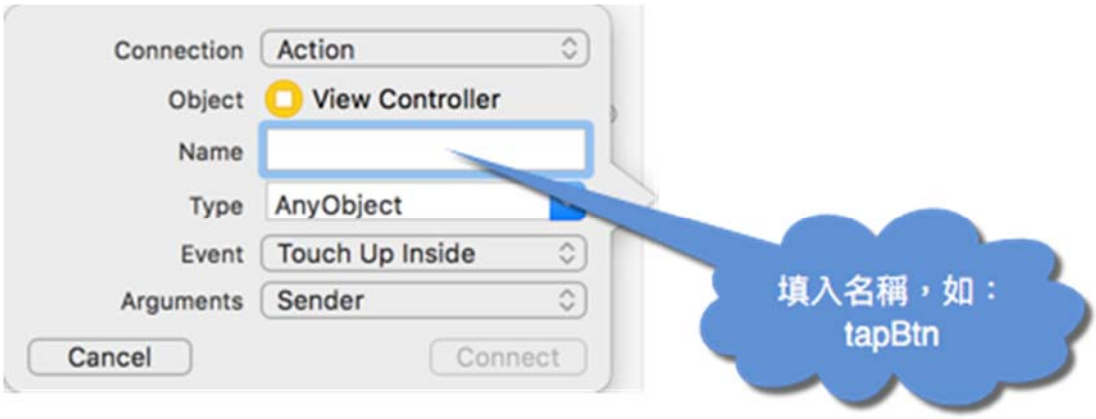


Step2. 滑鼠游標移至此元件上，按下滑鼠左鍵放開，即可點選此新增加的物件，再按住『Control』鍵，再次按下滑屬左鍵後，點選不放，拖拉滑鼠至右方的程式設計區，可以看到產生出一條藍色的線條，將滑鼠移至程式區的下方『 』』之前 （也就是最外層 {}之內） 鬆開滑鼠。
另一方法是在此元件上點選滑鼠右鍵按住不放，在拖拉滑鼠至左方程式設計區。

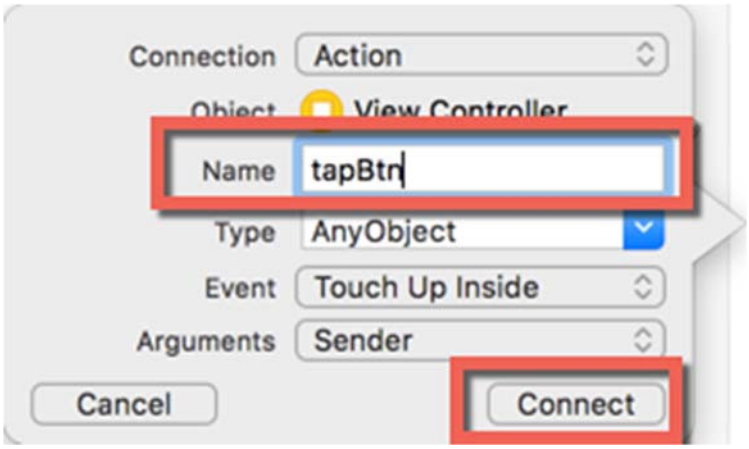


Step3. 鬆開滑鼠後，會產生一個對話視窗，此時可以於 Name 中填入此元件的名稱。
在本書中盡量 Action 的命名以動詞 tap 作為名稱開頭。
例如：Button 元件， 則命名 tapBtn。

在 event 中列出了多個可能的方法，一般若是點一下滑鼠就是使用 Touch Up Inside。



Step4. 當填入 tapBtn 於此元件的名稱後，按下 Connect。

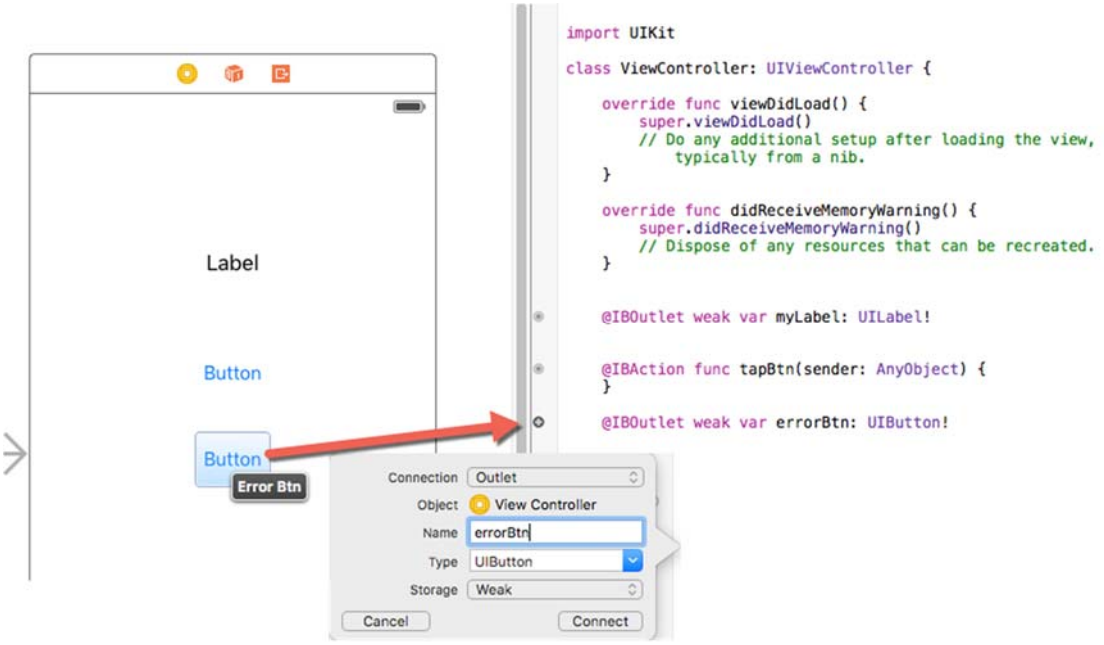


Step5. 在右方的程式設計區，將會自動插入一段程式碼。 此即完成了此 Button 的 Touch Up Inside 方法的命名。

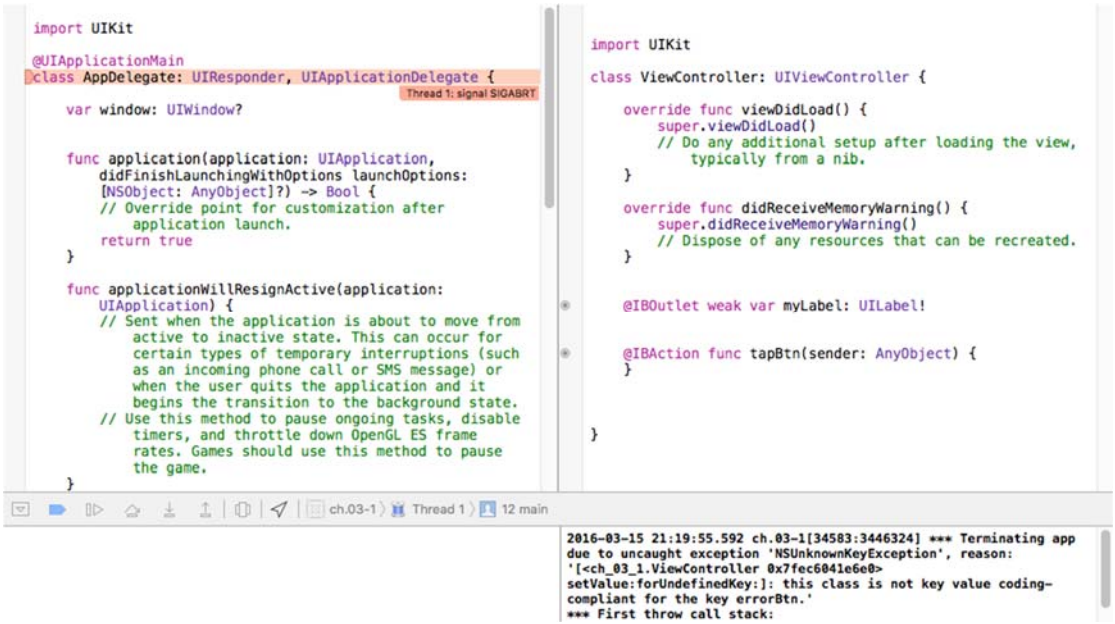
```
@IBAction func tapBtn(sender: AnyObject) {
    //填入程式碼
}
```

連結錯誤的處理方法

常常在建立 IBAction 時，因為一時不查未將 IBOutlet 改為 IBAction，舉例如下，建立了一個 errorBtn 的 IBOutlet。



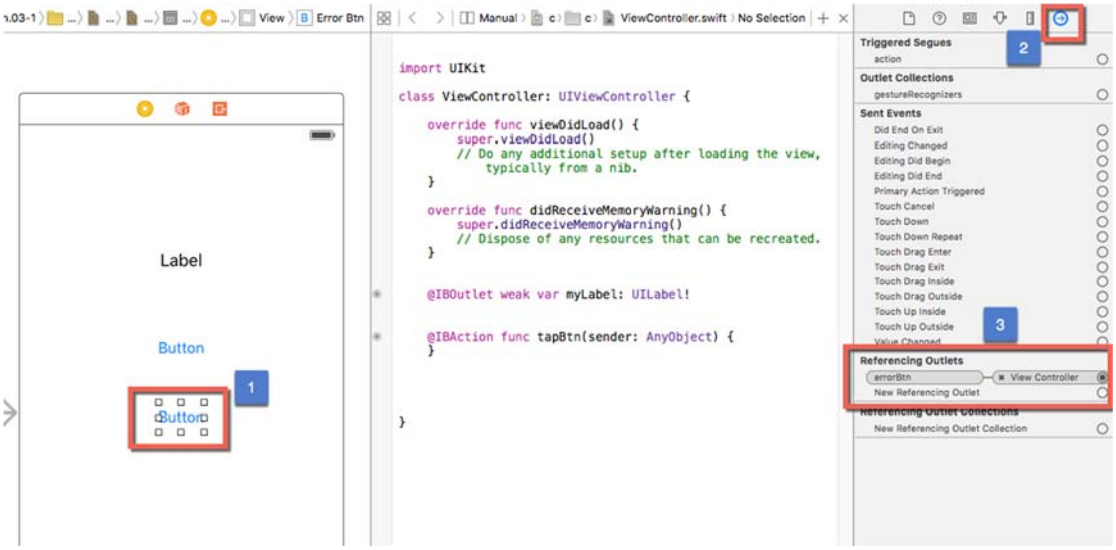
但事實上該是要建立一個 IBAction 才對，這時候切記不可只是將程式碼刪除到，若只是單獨將程式碼刪除，這時執行程式編譯時會出現錯誤，而且從這錯誤資訊中式無法判別錯誤原因的。



錯誤的原因在於雖然我們已經將錯誤的程式碼刪除

~~@IBOutlet weak var errorBtn: UIButton!~~

但是連結事實上並還未取消。 請點選此一新增的畫面元件，在專案右邊點選 Show the Connections inspector，可以發現到在下方有個 errorBtn 連結到 View Controller，此時請將 中間的 『x』 點選即可斷結連結。



接下來就可以將其設定為正確的 IBAction