

CHAPTER 使用者介面

得到與設定畫面上各個元件的位置與大小。



[問題]如何得到與設定畫面上各個元件的位置與大小

[解答]瞭解UIView的屬性，從中得知與設定畫面上各個元件的位置與大小

[範例程式碼]HelloUIView

[CGPoint、CGSize，以及CGRect]

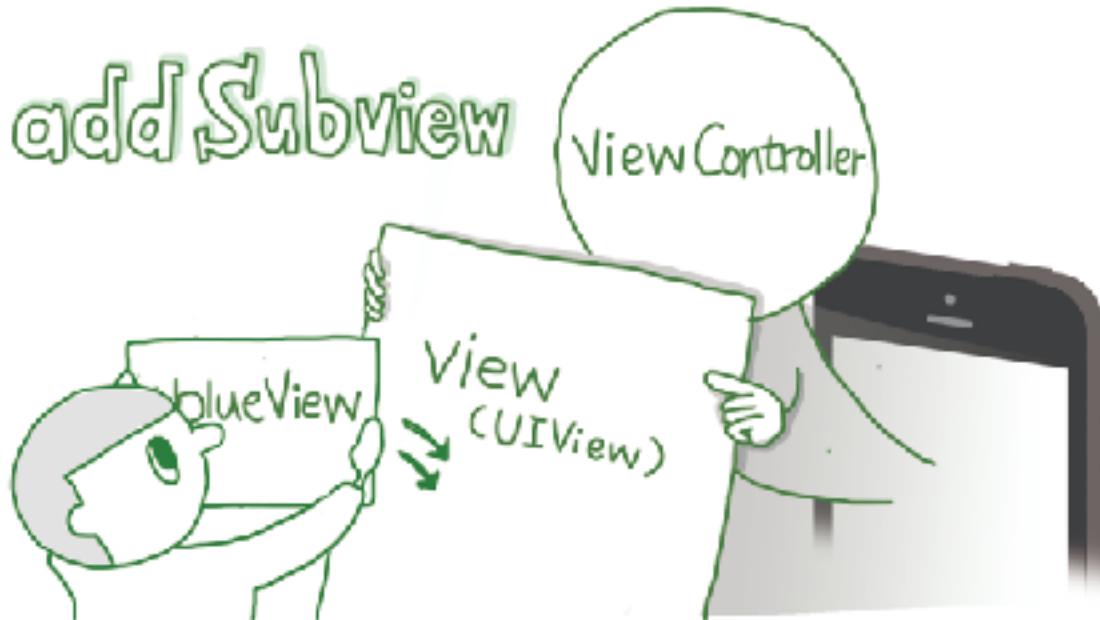
UIView是畫面上的一個區域。所有畫面上看到的元件都是UIView的子類別。比方說畫面上看到的圖片(UIImageView)、文字標籤UILabel)，或是按鈕UIButton)等，都是UIView的子類別。所以瞭解UIView的屬性與方法，就可以把UIView的屬性與方法應用到各個UIView的子類別中。瞭解如何得到與設定UIView的位置與大小，就能夠得到與設定畫面上各個元件的位置與大小。在介紹UIView之前，首先要瞭解三個重要的結構：

1. **CGPoint**：代表畫面上的一個點。可以用CGPoint的初始化方法產生出一個畫面上的座標。

2. **CGSize**：代表畫面上的一個區域。使用CGSize的初始化方法能夠產生出一個包含長度和寬度數值的區域。

3. **CGRect**：代表一個帶有座標的區域。使用CGRect的初始化方法能夠產生出一個包含x座標、y座標、寬度，與高度的CGRect。

[以程式碼產生UIView]



1. 請新開一個Single View Application專案，把ViewController.swift檔案裡 ViewController類別的程式碼，改成如下所示：

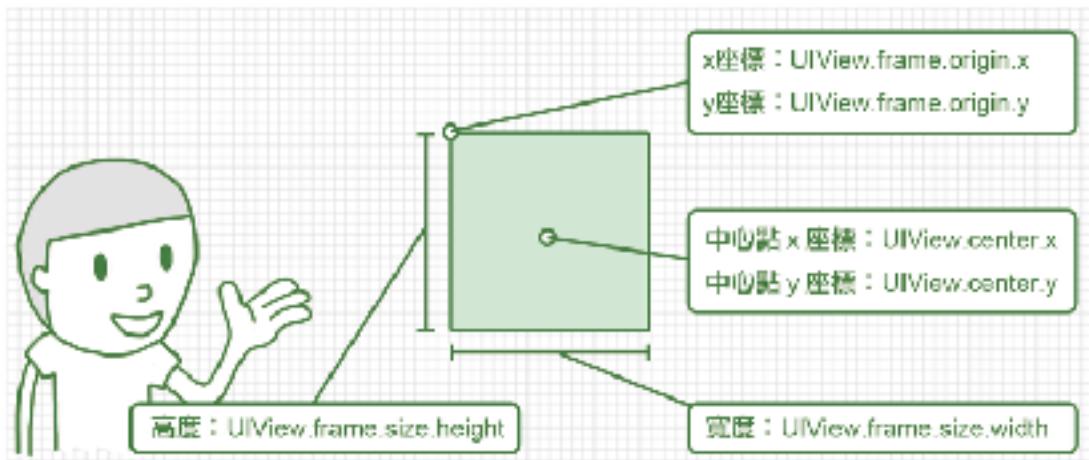
```
class ViewController: UIViewController {
    override func viewDidLoad() {
        super.viewDidLoad()
        let viewArea = CGRect(x: 50.0, y: 50.0,
                              width: 100.0, height: 100.0) //定義一個範圍
        var blueView = UIView(frame: viewArea) //使用範圍來產生一個UIView
        blueView.backgroundColor = UIColor.blue //設定顏色
        view.addSubview(blueView) //把blueView加到畫面上
    }
}
```

2. 上面的範例裡，試著在viewDidLoad方法裡面產生一個藍色的區域。(或者說產生一個藍色的UIView。)先呼叫CGRect的初始化方法定義UIView的範圍。這個方法需要四個參數：第一個參數是這個範圍在畫面上的x座標、第二個參數是此範圍在畫面上的y座標、第三個參數是該範圍的寬度，最後是該範圍的高度。

3. 使用步驟2的範圍產生UIView、設定此UIView的背景顏色。

4. 產生了藍色的UIView之後，要在畫面上顯示這個藍色的四方形區域的話，要呼叫addSubview方法，把blueView加到ViewController的畫面上。

【得到畫面上各個元件的位置與大小】



如果要得到畫面上各個元件的位置與大小，請參考下面的程式碼：

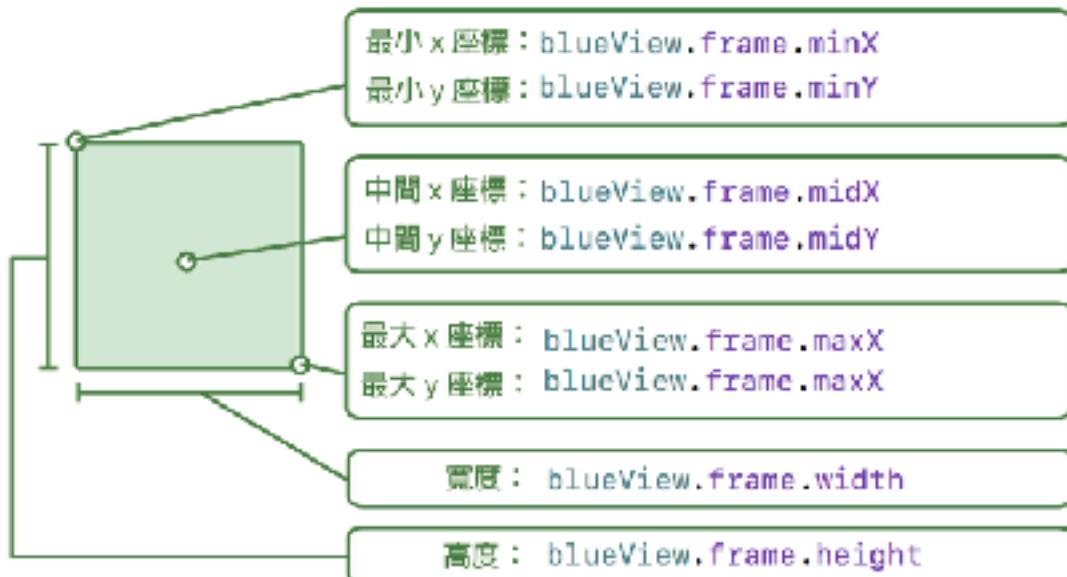
```

print(blueView.frame.origin.x)          //得到視圖的x座標
print(blueView.frame.origin.y)          //得到視圖的y座標
print(blueView.frame.size.width)        //得到視圖的寬度
print(blueView.frame.size.height)       //得到視圖的高度
print(blueView.center.x)               //得到視圖中心的x座標
print(blueView.center.y)               //得到視圖中心的y座標

```

- 1.想要得到畫面上視圖的x座標，請使用`UIView.frame.origin.x`；想要得到畫面上視圖的y座標，請使用`UIView.frame.origin.y`；想要得到畫面上視圖的寬度，請使用`UIView.frame.size.width`；想要得到畫面上視圖的高度，請使用`UIView.frame.size.height`。除此以外，也可以使用`UIView.center.x`與`UIView.center.y`得到畫面上視圖中心點的x與y座標。上面範例可以得到[以程式碼產生`UIView`]時，產生出來的`blueView`的各種資訊。
- 2.雖然程式碼是以`UIView`為例，不過由於`UIView`是眾多畫面上可視元件的父類別。於是`UIImageView`(圖片)、`UILabel`(文字標籤)...等，都可以用相同的方法，得到座標與其高度與寬度。

- 3.也請參考下面的方法：



也可以使用frame minX與frame minY得到畫面上某個區域的最小的x及y座標；使用frame midX與frame midY得到畫面上某個區域中間點的x及y座標；使用frame maxX與frame maxY得到畫面上某個區域的最大的x及y座標。除此以外，也可以使用frame width與frame height得到畫面上某區域的寬度與高度。

【設定畫面上各個元件的位置與大小】

想要設定畫面上各個元件的位置與大小，請參考下面的程式碼：

```
class ViewController: UIViewController {
    var blueView: UIView! //幫類別添加一個 UIView 屬性

    override func viewDidLoad() {
        //省略之前 blueView 的設定
    }

    override func viewDidAppear(_ animated: Bool) {
        //設定位置與大小的第一種方法
        blueView.frame.origin.x = 100
        blueView.frame.origin.y = 200
        blueView.frame.size.width = 50
        blueView.frame.size.height = 150

        //設定位置與大小的第二種方法
        let newPoint = CGPoint(x: 100, y: 200)
        blueView.frame.origin = newPoint
        let newSize = CGSize(width: 50, height: 150)
        blueView.frame.size = newSize

        //設定位置與大小的第三種方法
        let newArea = CGRect(x: 100, y: 200, width: 50, height: 150)
        blueView.frame = newArea
    }
}
```

範例中，把在[以程式碼產生UIView]單元中的blueView設定成類別的屬性。然後將設定UIView位置跟大小的程式碼寫在viewWillAppear方法裡面。以下是相關的說明：

1. 設定畫面視圖的大小與位置有好幾種方法，其中的第一種，就是直接透過UIView.frame.origin.x、UIView.frame.origin.y、UIView.frame.size.width，與UIView.frame.size.height設定視圖的位置與大小；第二種方法是建立CGPoint與CGSize，透過CGPoint調整視圖的位置，透過CGSize調整視圖的大小；第三種是建立同時含有座標與大小的CGRect，用CGRect來設定。
2. 雖然程式碼是以UIView為例，不過由於UIView是眾多畫面上可視元件的父類別。於是UIImageView(圖片)、UILabel(文字標籤)...等，都可以用相同的方法，設定座標與其高度與寬度。
3. 設定與得到視圖大小與位置的程式碼，是寫在viewDidAppear方法裡面，也可以寫在viewDidLayoutSubviews。這些程式碼沒有寫在viewDidLoad方法裡面，是因為程式在執行viewDidLoad方法時，尚未完全擺放好各個視圖的位置。

目前執行程式機種畫面的寬度與高度

[問題]如何得到目前執行程式機型畫面的寬度與高度

[解答]瞭解UIView的屬性，從中得知不同機型畫面的寬度與高度

[範例程式碼]HelloCheckWidthAndHeight

[過程解說]

根據上個問題的解答，懂得如何偵測UIView的寬度跟高度之後，想要知道目前執行程式機型畫面的寬度與高度請參考下面的程式碼：

```
class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        print(view.frame.size.width) //得到畫面的寬度
        print(view.frame.size.height) //得到畫面的高度
    }
}
```

打開新的專案後，在ViewController類別中，寫下view.frame.size.width就可以得到目前執行程式機型畫面的寬度；而使用view.frame.size.height就可以得到畫面的高度。得到畫面的高度與寬度在某些情況中很重要，可以依照不同的數值在不同大小的畫面中擺放新的元件。

使用文字標籤顯示文字訊息UILabel)



[問題]如何在畫面上顯示文字訊息

[解答]使用文字標籤UILabel顯示文字訊息

[範例程式碼]HelloUILabel

[過程解說]

1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UILabel、用滑鼠把文字標籤拉到畫面上。

1. 選 Main.storyboard



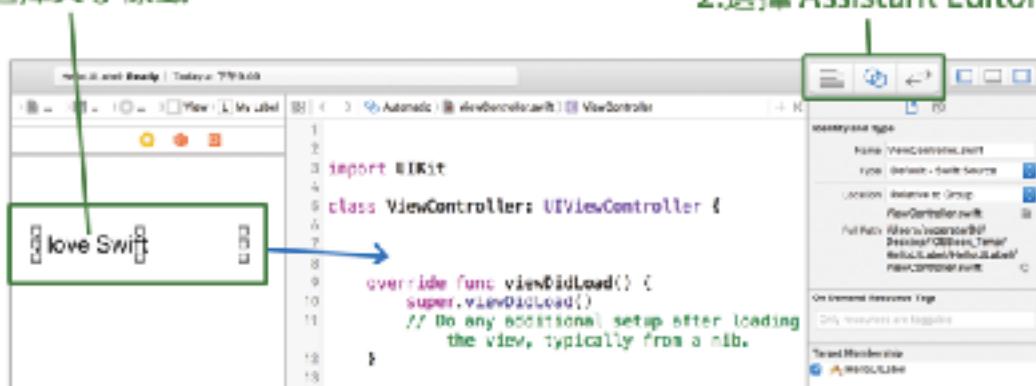
2. 點選步驟1加入的文字標籤，可以在右邊欄做出各種相關設定。



[以程式碼改動文字內容]

如果要做到以程式碼改動UILabel文字內容的話，請連結UILabel到程式碼中。

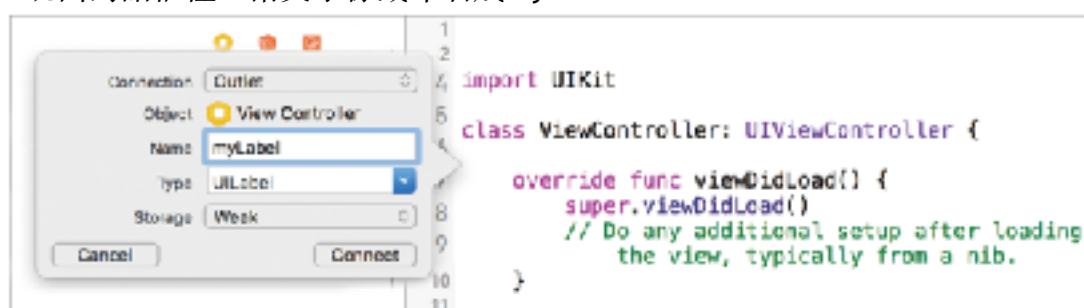
1. 選擇文字標籤



3. 一面按著鍵盤的 control 鍵，一面連結文字標籤到程式碼中

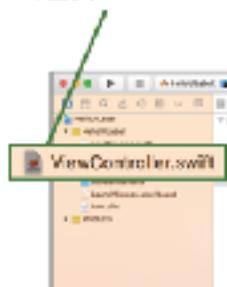
1. 請先選擇文字標籤，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結文字標籤到程式碼中。

2. 跳出對話框裡，幫文字標籤命名成myLabel。

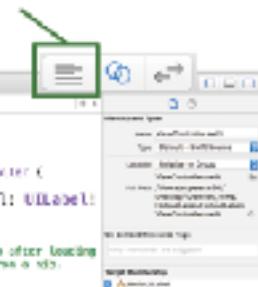


3. 您可以幫您的文字標籤命名成其他的名稱，範例中命名成myLabel。之後就以myLabel這個屬性名稱代表畫面上的文字標籤。連結完成之後，請按Standard Editor，回到單一畫面。之後請再選擇左邊欄的ViewController.swift檔案。

2. 選擇ViewController.swift



1. 連結完畢之後按下Standard Editor



4. 請更改ViewController類別中的程式碼：

```
override func viewDidLoad() {
    super.viewDidLoad()
    myLabel.text = "Hello! Swift!" //更改文字內容
    myLabel.textColor = UIColor.red //更改文字顏色
    myLabel.textAlignment = .center //對齊方式
    myLabel.backgroundColor = UIColor.yellow //底色
    myLabel.font = UIFont(name: "Arial", size: 24) //字型
}
```

5. 可以用UILabel的text屬性更改文字內容；textColor屬性更改文字顏色；textAlignment屬性更改文字對齊方式；backgroundColor屬性更改文字的底色，以及使用font屬性更改文字字型與文字大小。用這樣的方式就可以用程式碼更改文字標籤的內容。

6. 文字對齊的方式除了範例中的「.center(置中對齊)」以外，還有「.left(靠左對齊)」、「.right(靠右對齊)」、「.justified(左右對齊)」，以及「.natural(依照語言情況不同做不同的對齊)」。

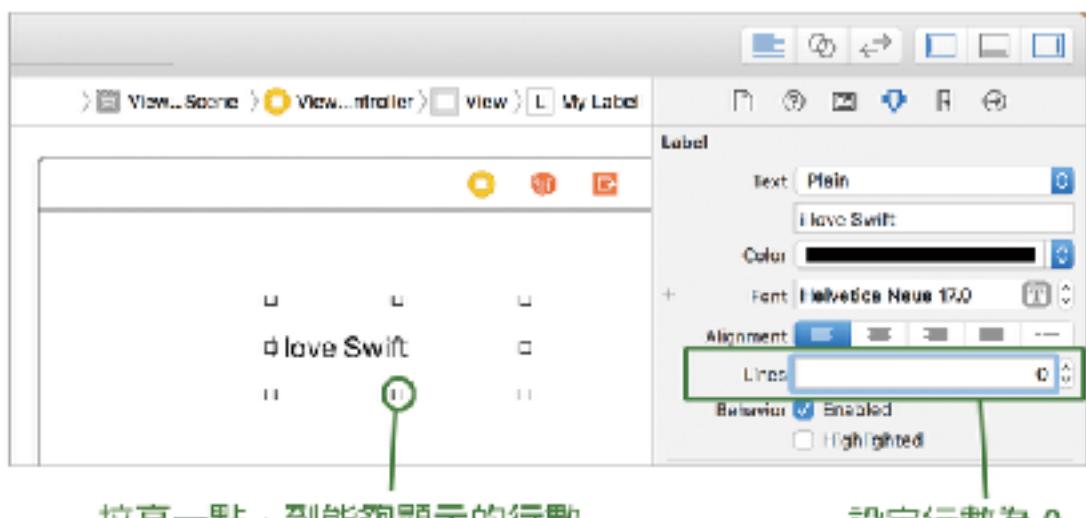
【以程式碼建立文字標籤】

範例程式碼同時記錄了如何直接用程式碼生成UILabel。如果要使用程式碼生成文字標籤的話，請參考本單元的範例程式碼。

【用文字標籤多行文字】



1. 通常顯示多行文字會使用文字方塊(Text View)。不過如果要用文字標籤(UILabel)顯示多行文字的話，請回到Main.storyboard檔案，選擇文字標籤。在行數設定的欄位中，填入0。
行數設定填入1，是設定文字標籤的行數為1行；行數設定填入1，是設定文字標籤的行數為2行；依此類推，不過如果填入的行數為0的話，代表不限制行數。

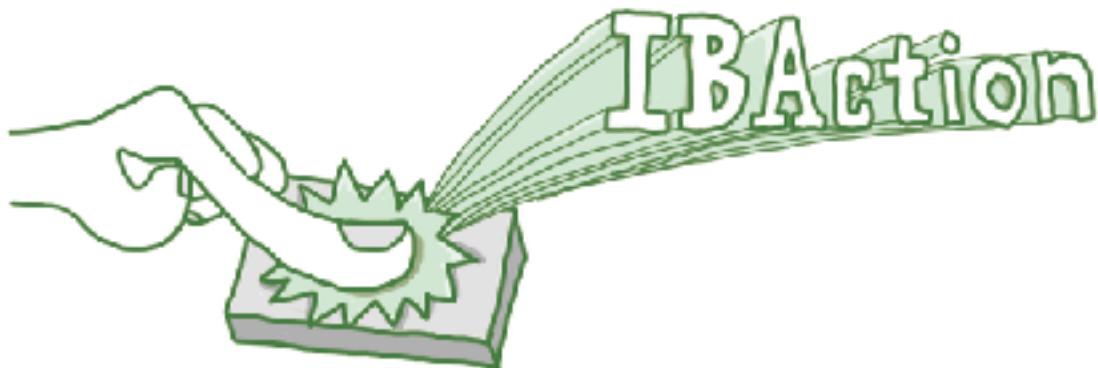


2. 在上個範例的viewDidLoad方法中，把顯示文字改成如下所示：

```
myLabel.text = "Hello\nWorld"
```

字串中「\n」代表換行(new line)，把文字改成上面的字串的話，會在顯示「Hello」之後換行，在新的一行開頭顯示「World」。

使用按鈕(UIButton)



[問題]如何讓畫面上的按鈕按下後會執行設定的功能

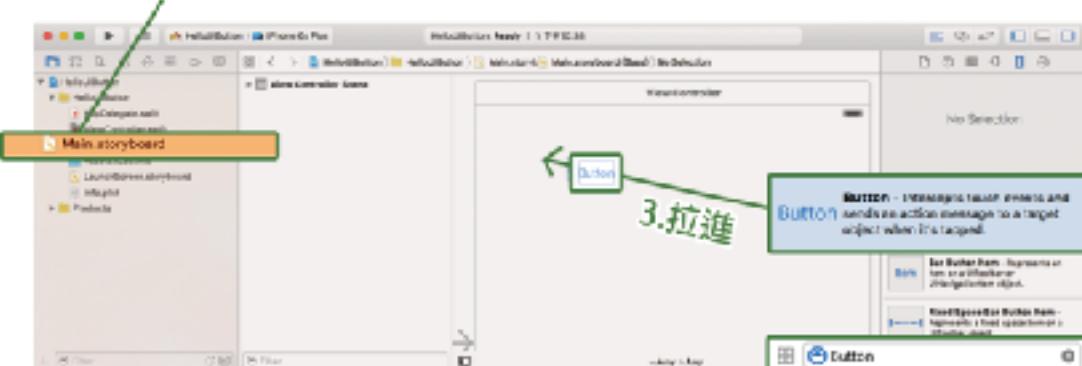
[解答]連結Storyboard上的按鈕到程式碼中，做後續的設定

[範例程式碼]HelloUIButton

[過程說明]

1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UIButton、用滑鼠把按鈕拉到畫面上。

1. 選 Main.storyboard

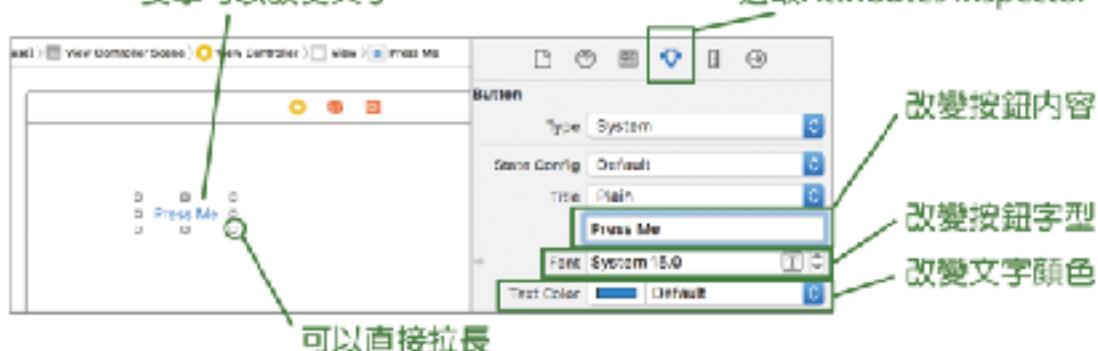


2. 提尋 UIButton

2. 點選步驟1加入的按鈕，可以在右邊欄做出各種相關設定。

雙擊可以改變文字

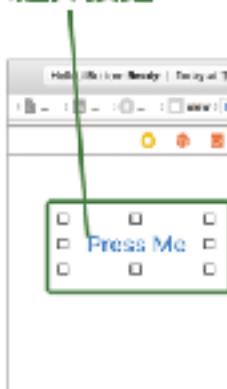
選取Attributes Inspector



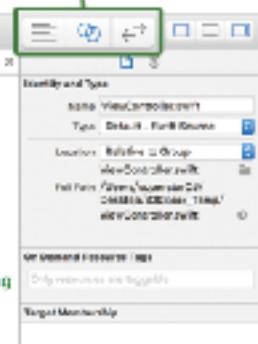
【使用按鈕】

如果要讓畫面上的按鈕按下後會執行設定的功能，請連結UIButton到程式碼中。

1. 選擇按鈕

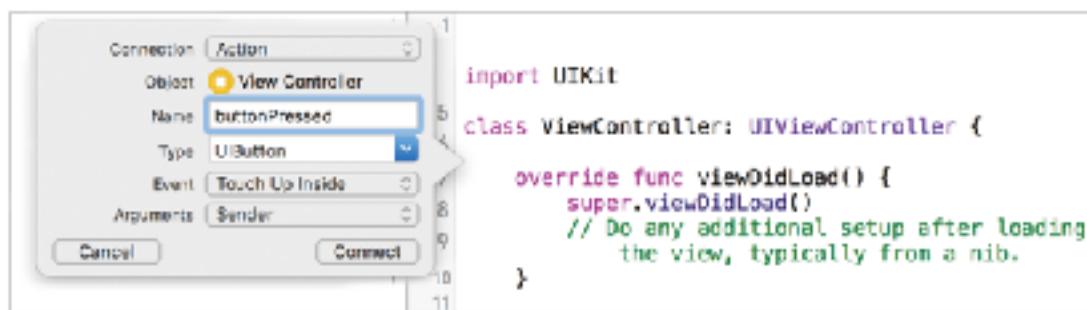


2. 選擇 Assistant Editor



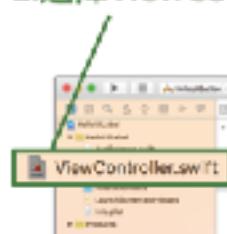
3. 一面按著鍵盤的 control 鍵，一面連結按鈕到程式碼中

1. 請先選擇按鈕，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結按鈕到程式碼中。
2. 跳出對話框裡，在Connection選擇Action、幫這個按鈕觸發的事件(IBAction)命名成buttonPressed，並把Type選擇成UIButton。

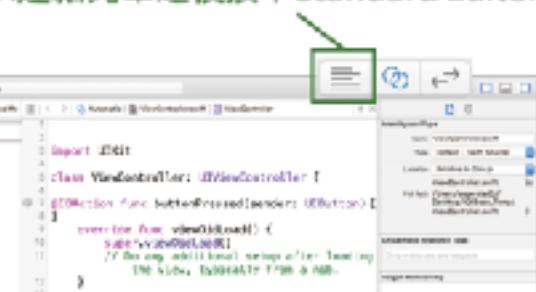


3. 您可以幫按鈕觸發的事件命名成其他的名稱，範例中命名成buttonPressed。之後就以buttonPressed這個方法名稱代表按鈕觸發的事件。連結完成之後，請按Standard Editor，回到單一畫面。之後請再選擇左邊欄的ViewController.swift檔案。

2. 選擇ViewController.swift



1. 連結完畢之後按下Standard Editor



4. 在ViewController類別的buttonPressed方法中寫下程式碼：

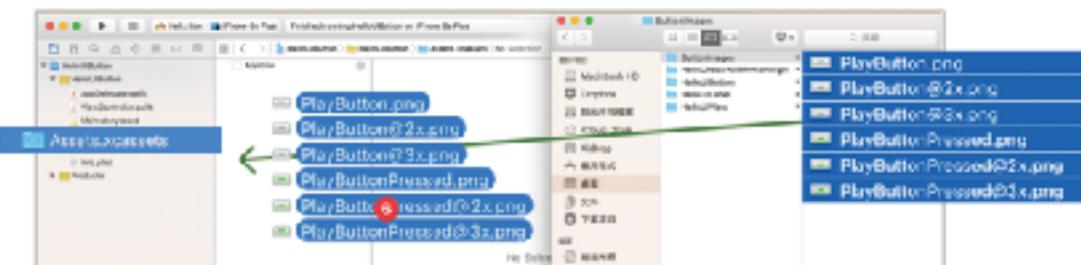
```
class ViewController: UIViewController {
    @IBAction func buttonPressed(_ sender: UIButton) {
        print("button pressed")
    }
}
```

5.範例程式碼中，設定按下按鈕之後，會印出「button pressed」的除錯文字。您可以依照情況，在對應的方法中，加入想要執行的工作。

[設定按鈕圖片]



1.除了把無外框的文字當成按鈕以外，也可以設定使用圖片來當作按鈕的外觀。本範例會使用到圖檔。這些圖檔在隨書附送的程式碼資料夾裡，請把圖片都匯入到專案中。(圖片於ButtonImages資料夾)

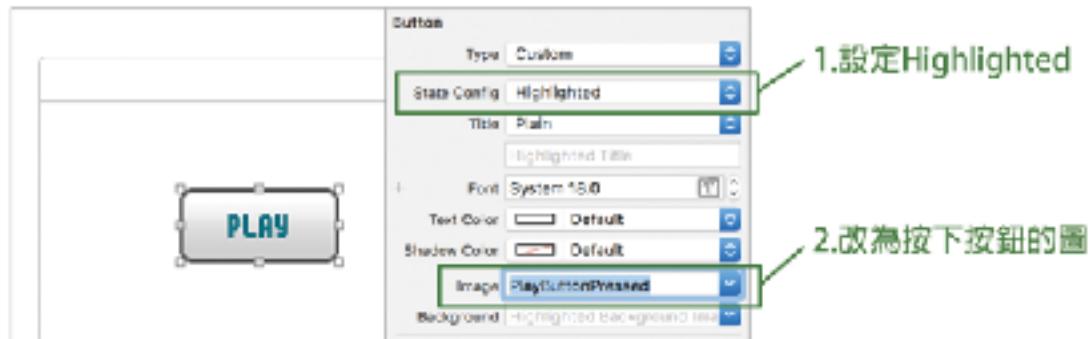


2.選取Main.storyboard，選取按鈕，刪除原本按鈕的文字，然後在Image的欄位設定圖片。



3.上個步驟是設定還沒有按下按鈕、預設Default狀態要顯示的圖片。選擇State

Config欄位，更改設定為Highlighted後，就可以設定按下去要顯示的圖片。以此方法就可以設定以客製化的圖片來當作按鈕。



【使用程式碼製作按鈕】

範例程式碼同時記錄了如何直接用程式碼生成顯示文字或顯示圖片的UIButton。如果要使用程式碼生成按鈕的話，請參考本單元的範例程式碼。

使用文字輸入框來輸入資料 (UITextField)



[問題]如何使用文字輸入框來讓使用者輸入資料

[解答]連結Storyboard上的文字輸入框到程式碼中，做後續的設定

[範例程式碼]HelloUITextField

[過程說明]

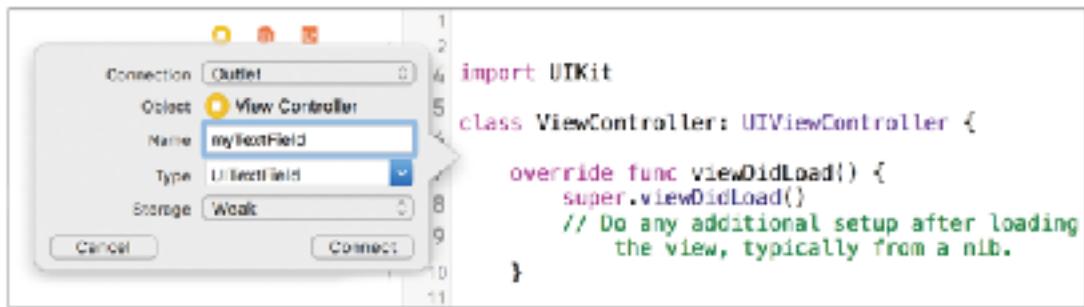
1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UITextField、用滑鼠把文字輸入框拉到畫面上。



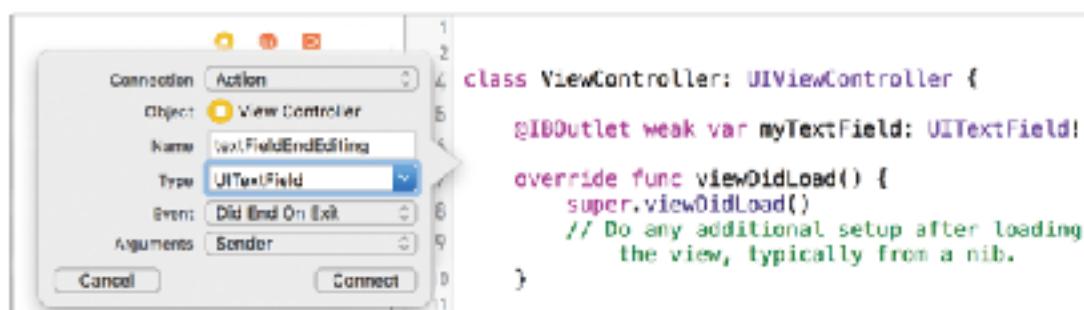
2. 點選步驟1加入的文字輸入框，可以在右邊欄做出各種相關設定。其中比較特別的，是Placeholder這個欄位。在這邊Placeholder填入文字，會在畫面中以淡灰色的方式呈現，提示使用者在文字輸入框中輸入訊息。



3. 如果要得知使用者在文字輸入框輸入的資料，要將文字輸入框連結到程式碼中。請先選擇文字輸入框，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結文字輸入框到程式碼中。跳出的對話框裡，幫文字輸入框命名成myTextField。



4.重複步驟3的動作，再次連結文字輸入框到程式碼中。只不過這次跳出對話框裡，在Connection選擇Action、幫這個文字輸入框觸發的事件(IBAction)命名成textFieldEndEditing，並把Type選擇成UITextField。連結完成後，使用者輸入資訊結束、按下畫面虛擬鍵盤上的return鍵會觸發textFieldEndEditing方法。



4.回到程式碼，在ViewController類別裡的textFieldEndEditing方法中，寫入下面的程式碼：

```

class ViewController: UIViewController {
    @IBOutlet weak var myTextField: UITextField!
    @IBAction func textFieldEndEditing(sender: UITextField) {
        print(myTextField.text!)      //印出文字輸入框的文字
        myTextField.text = ""        //將文字輸入框清空
    }
}

```

5.程式碼中，myTextField代表畫面上的文字輸入框，使用者在點擊文字輸入框、輸入資訊後按下虛擬鍵盤的return鍵、觸發textFieldEndEditing方法時，可以用myTextField.text得到文字輸入框中的文字。在後面加上驚嘆號的意思是，確認該屬性一定有值。範例中，印出使用者輸入的文字，隨即設定myTextField.text為空字串，清空文字輸入框，讓使用者可以再次輸入新的資訊。

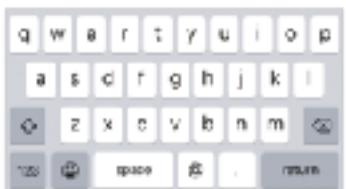
[各種鍵盤樣式]



Default



NumberPad



E-mail Address

依照各種情況，可以設定對應的鍵盤樣式：在設定文字輸入框的Attributes Inspector欄下，調整Keyboard Type，可以讓跳出來的鍵盤符合各種使用情況。預設鍵盤(Default)是文字輸入；數字鍵盤(NumberPad)是專門輸入數字；電子郵件鍵盤(E-mail Address)加入了「@」與「.」這些撰寫電子郵件常用的符號。

[輸入完成按鈕顯示文字]

虛擬鍵盤上的return鍵也可以依照各種不同情況做不同的設定：在設定文字輸入框的Attributes Inspector欄下，調整Return Key，可以讓虛擬鍵盤上的return鍵顯示不同的文字，以下是除了預設(Default)的Return文字以外，其他可能的選項：

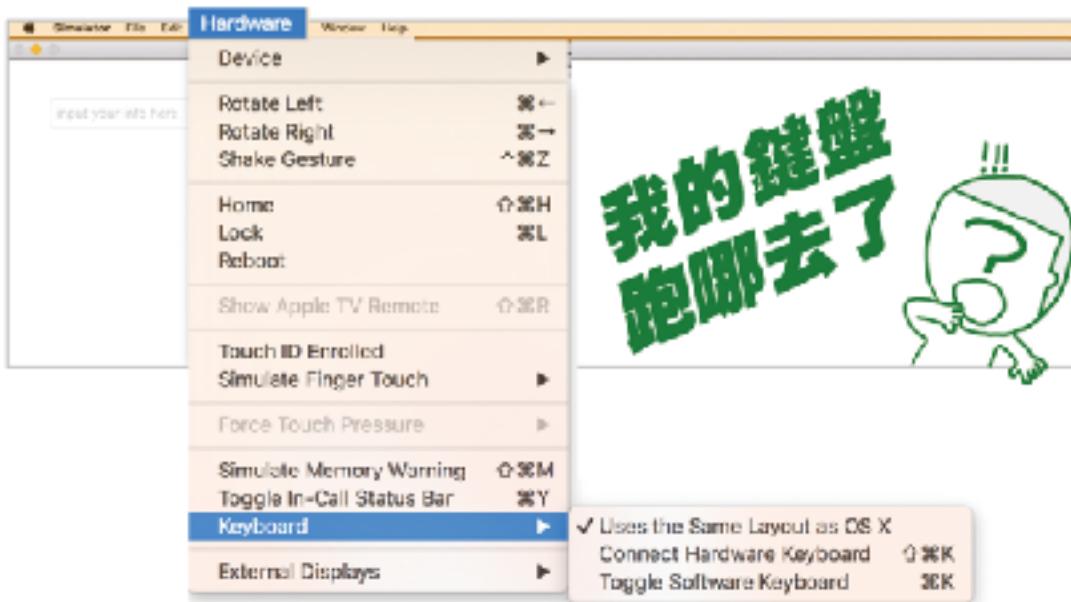
Go	Google	Join	Next	Route
Search	Send	Yahoo	Done	Emergency Call

[顯示清除鈕 (Clear Button)]



文字輸入框靠進右邊邊界，可以設定打叉的按鈕。按下去可以清除使用者輸入的文字。這個按鈕叫做「清除鈕(Clear Button)」，預設不會顯示。如果要顯示此按鈕，請在設定文字輸入框的Attributes Inspector欄下的Clear Button選項，調整成「is always visible」會在任何情況都會顯示Clear Button；調整成「Appear while editing」則是在輸入過程中才會出現此按鈕。

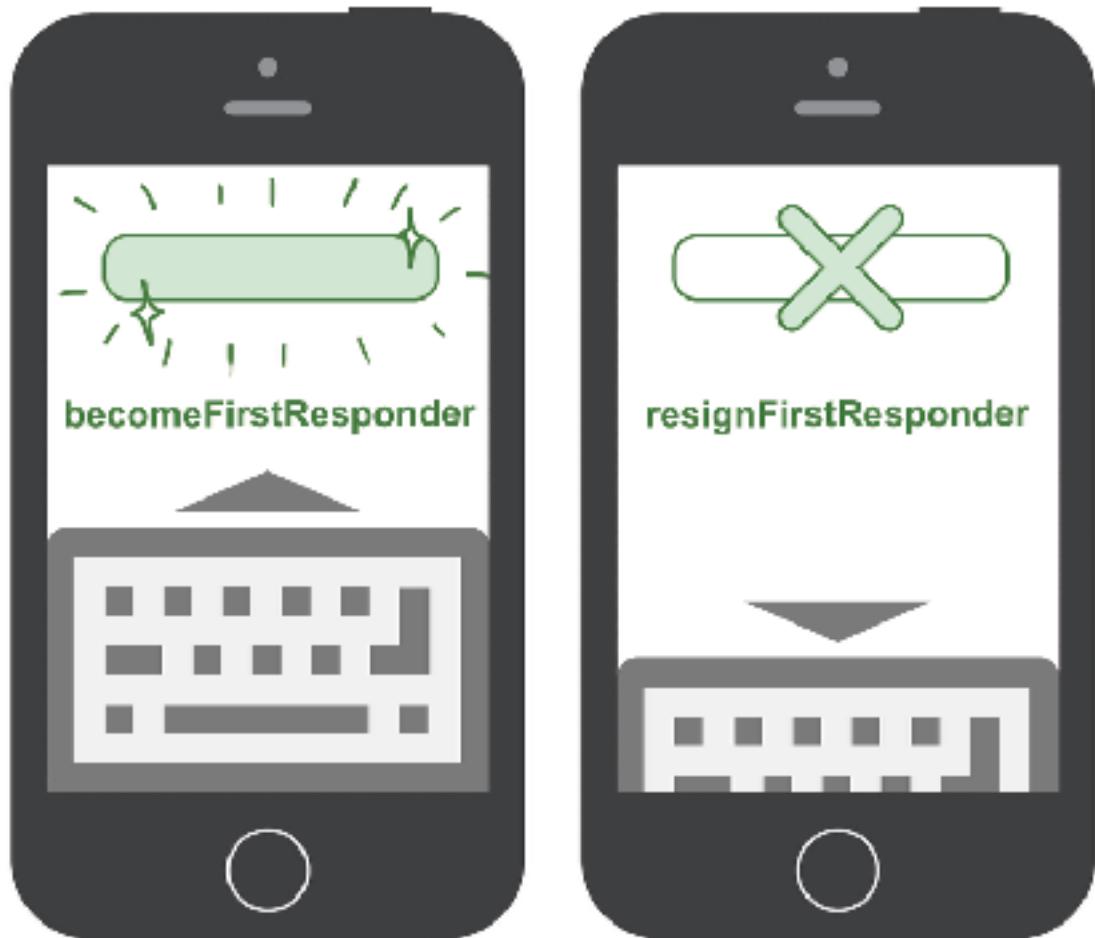
[模擬器執行鍵盤無法跳出]



開發程式的過程中，有時候會發現模擬器鍵盤無法跳出。通常是因為不小心選到連接到了實體鍵盤的選項。請照著下面的步驟排除錯誤。

1. 選擇模擬器
2. 選擇上面工具列中的[Hardware] > [Keyboard]，取消勾選「Connect Hardware Keyboard」。模擬器的虛擬鍵盤就會重新跳出。

【鍵盤自動跳出】



預設文字輸入框的行為，需要使用者點擊之後，才會跳出鍵盤，供使用者輸入資料。如果想在畫面剛剛呈現在使用者面前、使用者還沒點擊文字輸入框就跳出鍵盤的話，請在ViewController類別的viewDidLoad的方法裡面，輸入下面的程式碼：

```
override func viewDidLoad() {  
    myTextField.becomeFirstResponder()  
}
```

呼叫文字輸入框的becomeFirstResponder方法，可以讓文字輸入框變成畫面上的焦點、而鍵盤就會在畫面剛剛呈現在使用者面前、使用者還沒點擊文字輸入框時跳出。

【按下畫面任何一個地方收起鍵盤】

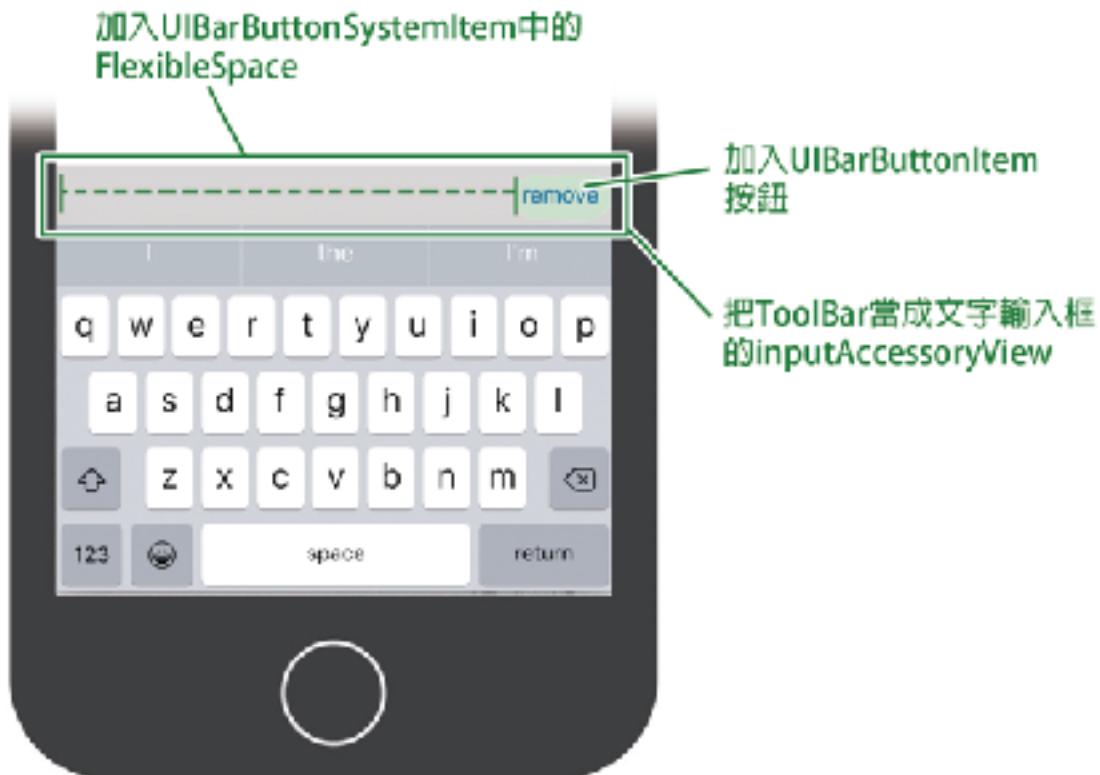
預設文字輸入框的行為，需要使用者輸入資訊、點擊虛擬鍵盤的return鍵，鍵盤才會收回。如果想要做到輸入資訊到一半、點擊畫面上非鍵盤的任何地方，就能夠收起鍵盤的功能，請在ViewController類別裡覆寫下面和觸碰相關的方

法：

```
override func touchesBegan(_ touches: Set<UITouch>,
                           with event: UIEvent?) {
    myTextField.resignFirstResponder()
}
```

程式碼的意思是，剛碰到畫面的任何地方時，呼叫文字輸入框的 `resignFirstResponder` 方法。這樣鍵盤就會收起、需要使用者再次點擊文字輸入框，虛擬鍵盤才會重新跳出。

【在鍵盤上方加入工具列與按鈕】



如果想幫文字輸入框跳出來的鍵盤加上工具列與按鈕的話，請參考下面的程式碼：

```

override func viewDidLoad() {
    //產生toolBar
    let myToolBar = UIToolbar(frame:
        CGRect(x: 0, y: 0, width: view.frame.width, height: 44))
    //產生barButtonItem按鈕
    let removeButton = UIBarButtonItem(title: "remove",
        style: .plain, target: self,
        action: #selector(ViewController.removeKeyboard))
    //產生flexible space
    let someSpace =
        UIBarButtonItem(barButtonSystemItem: .flexibleSpace,
                        target: nil, action: nil)
    //把彈性空間和按鈕加到 toolBar 中
    myToolBar.setItems([someSpace,removeButton], animated: false)
    //設定toolBar為文字輸入框的 inputAccessoryView
    self.myTextField.inputAccessoryView = myToolBar
}

func removeKeyboard(){
    myTextField.resignFirstResponder()
}

```

- 1.先產生工具列(ToolBar)。
- 2.生成放在工具列上的按鈕，以及排版所需的彈性空間。
- 3.把彈性空間與按鈕擺放到ToolBar上。
- 4.設定工具列為文字輸入框的inputAccessoryView，就可以在鍵盤上方加上工具列與按鈕。
- 5.範例中設定按下按鈕會執行removeKeyboard方法，在removeKeyboard方法中呼叫文字輸入框的resignFirstResponder方法收起鍵盤。

[使用程式碼製作文字輸入框]

範例程式碼同時記錄了如何直接用程式碼生成文字輸入框。如果要使用程式碼生成文字輸入框的話，請參考本單元的範例程式碼。

文字方塊顯示大量文字(UITextView)



[問題]如何文字方塊顯示大量文字

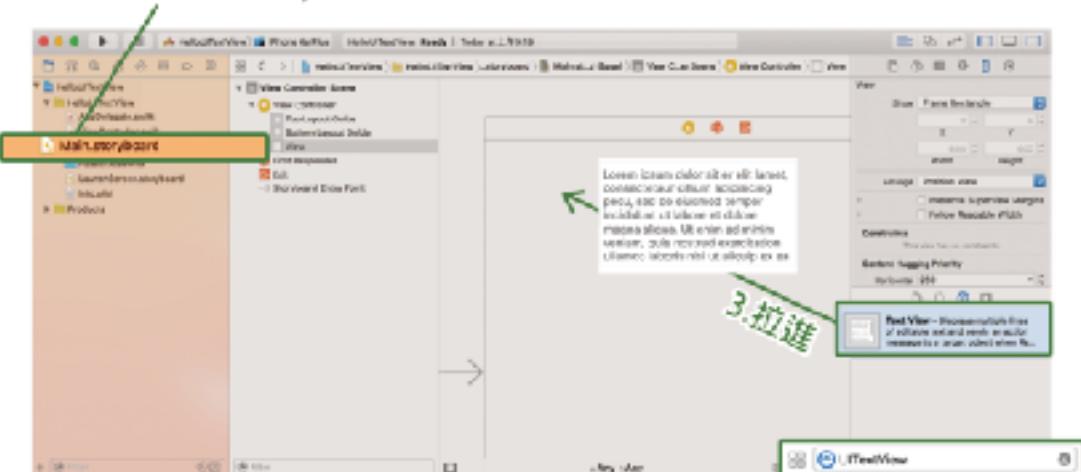
[解答]設定 UITextView 的 text 屬性，可以確定其顯示的文字

[範例程式碼]HelloUITextView

[過程說明]

1. 請先新增一個 Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋 UITextView、用滑鼠把文字方塊拉到畫面上。

1. 選 Main.storyboard



2. 搜尋 UITextView

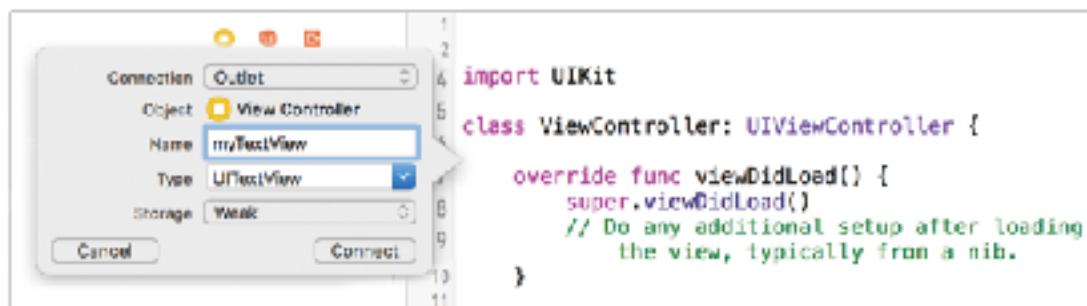
2. 點選步驟1加入的文字方塊(UITextView)，可以在右邊欄做出各種相關設定。可以直接輸入文字方塊要顯示的文章。開啟Editable功能的話，使用者點選到文字方塊就會跳出鍵盤，供使用者修改文字方塊中的內容。開啟Selectable選項的話，使用者就可以選取文字方塊中的文字。想要顯示大量文字的話，可以使用UITextView呈現。



【使用程式碼更改文字方塊顯示的內容】

如果要使用程式碼更改文字方塊顯示的內容，要將文字方塊連結到程式碼中。請依照下面的流程設定：

1. 先選擇文字方塊，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結文字方塊到程式碼中。跳出的對話框裡，幫文字輸入框命名成myTextView。



2. 回到程式碼，在ViewController類別裡的viewDidLoad方法中，寫入下面的程式碼：

```

override func viewDidLoad() {
    super.viewDidLoad()
    myTextView.text = "驢王子\n\n很久很久以前，在歐洲有一個小王國。這個國家的  
國王和皇后很想要生一個小孩，不過怎麼都生不出來。他們每天都向天祈求，希望能夠生出孩子。\n某日天神出現了，他答應國王和皇后，即將成全他們的願望。\n不久皇后變懷孕了，沒想到生出來的不是小嬰兒，而是頭小驢子..."}
}

```

3. 程式碼中，myTextView代表畫面上的文字方塊，使用myTextView.text屬性就可以設定文字方塊顯示的文字。碰到要斷句換行的時候，使用跳脫字元「\n」換行。用這樣的方法就可以在程式碼中更改文字方塊顯示的內容。

【結束編輯文字方塊】



文字方塊在Storyboard設定成「可編輯(Editable)」時，點選文字輸入框會跳出鍵盤供使用者修改內容。這時會發現，文字方塊的鍵盤按下return鍵只會換行，並不會收起鍵盤結束編輯。如果想要結束編輯內容的話，請依照文字輸入框單元的最後、在[鍵盤上加入工具列與按鈕]的內容，讓使用者按下按鈕，改成觸發「myTextView.resignFirstResponder()」，就能夠成功地收起鍵盤。

【使用程式碼製作文字方塊】

範例程式碼同時記錄了如何直接用程式碼生成文字方塊。如果要使用程式碼生成文字方塊的話，請參考本單元的範例程式碼。

可切換On/Off的開關(UISwitch)



【問題】如何製作開關

【解答】設定UISwitch，其屬性on決定目前開關狀態

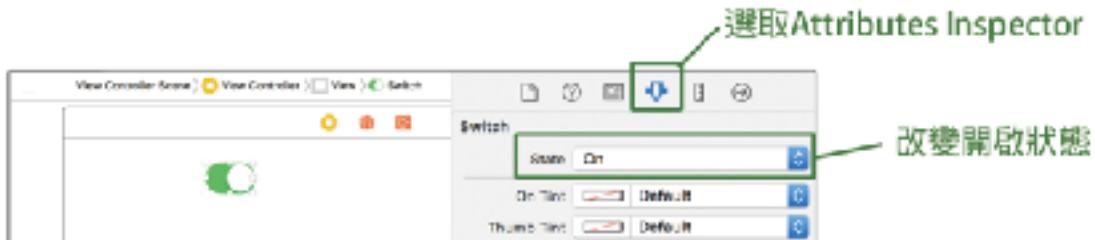
【範例程式碼】HelloUISwitch

【過程說明】

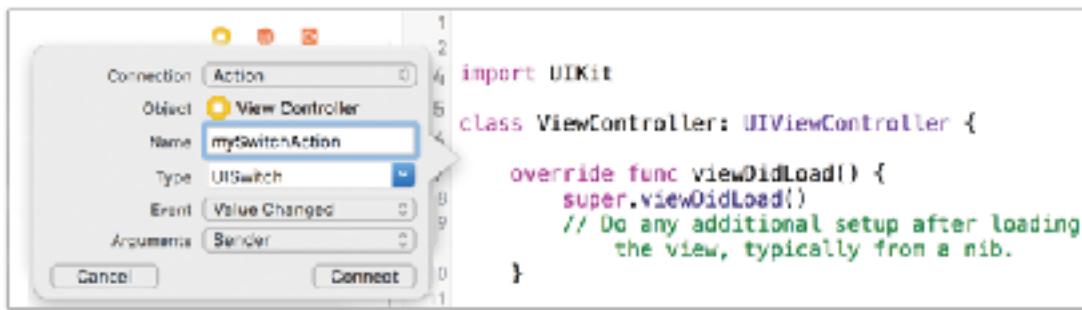
1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UISwitch、用滑鼠把開關拉到畫面上。



2. 點選步驟1加入的開關(UISwitch)，可以在右邊欄直接設定開關狀態。



3. 如果想要接受使用者開關UISwitch的事件，請將UISwitch連結到程式碼中：選擇UISwitch，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UISwitch到程式碼中。跳出的對話框裡，請先選擇Action，接著幫這個開關會觸發的方法命名成mySwitchAction。Type一欄請選UISwitch，event選Value Changed。
設定完成後，使用者撥動開關，讓開關狀態改變時(Value Changed)，就會觸發 mySwitchAction方法。



4.回到程式碼，在ViewController類別裡的mySwitchAction方法中，寫入下面的程式碼：

```
@IBAction func mySwitchAction(_ sender: UISwitch) {  
    if sender.isOn == true{  
        print("Switch is now on") //目前開關是開啟  
    }else{  
        print("Switch is now off") //目前開關是關閉  
    }  
}
```

5.程式碼中，可以使用觸發該方法的sender、也就是開關的屬性isOn判斷目前開關的狀態：如果屬性isOn的值是true的話，代表開關的狀態是開啟；相反則代表關閉。

【使用程式碼製作開關】

範例程式碼同時記錄了如何直接用程式碼生成開關(UISwitch)。如果要使用程式碼直接生成開關的話，請參考本單元的範例程式碼。

使用滑桿(UISlider)

UISlider.value



[問題]如何使用滑桿

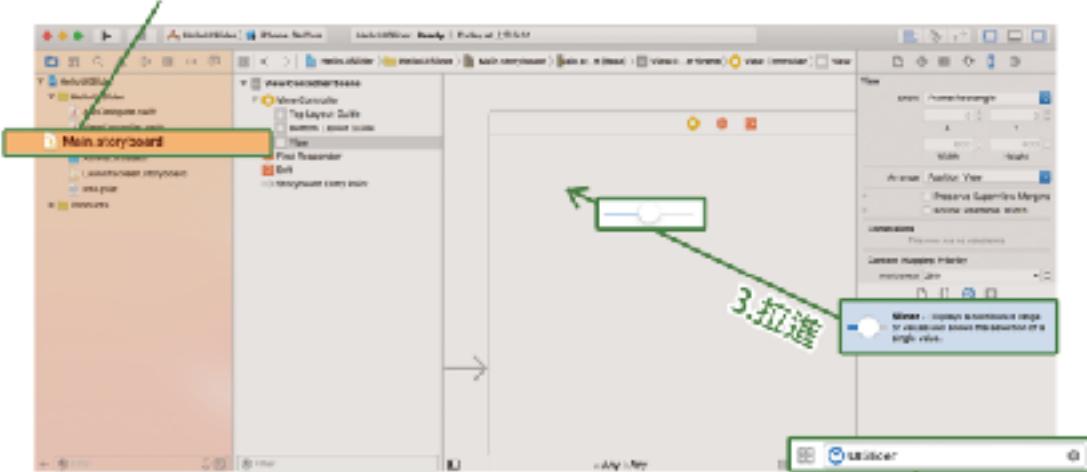
[解答]設定UISlider，從其屬性value得到目前滑桿的滑動的位置

[範例程式碼]HelloUISlider

[過程說明]

- 1.請先新增一個Single View Application。新增之後，在左邊欄點選Main.storyboard。然後在右下方搜尋UISlider、用滑鼠把滑桿拉到畫面上。

1.選 Main.storyboard



2.搜尋 UISlider

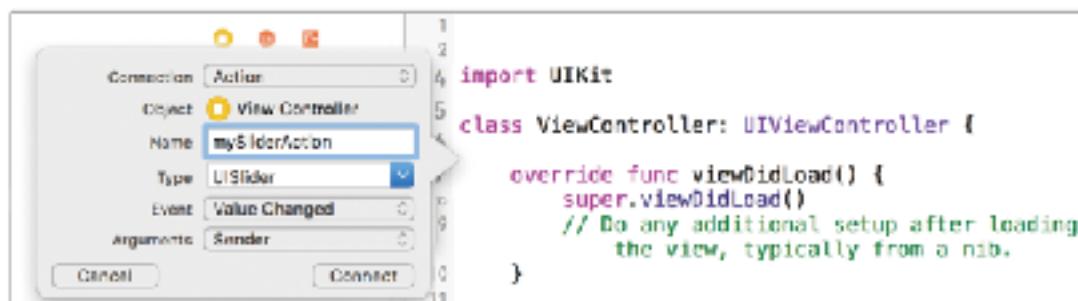
- 2.點選步驟1加入的滑桿(UISlider)，可以在右邊欄設定滑桿的最小值、最大值，與目前的數值。預設最小值是0、最大值是1，預設數值為0.5。不過在此範例中，把最小值調成1，最大值調成100，把預設數值調成50。
右邊欄中還有「Continuous Update」的選項。如果開啟此選項，會在使用者移動滑桿的過程中持續不停地更新滑桿的數值；相反的，如果關閉此選項，則只

會在使用者移動滑桿後更新當時滑桿的數值。



3.如果想要接受使用者移動UISlider的事件，請將UISlider連結到程式碼中：選擇UISlider，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結滑桿到程式碼中。跳出的對話框裡，請先選擇Action，接著幫這個開關會觸發的方法命名成mySliderAction。Type一欄請選UISlider，event選Value Changed。

設定完成後，使用者滑動UISlider，讓滑桿狀態改變時(Value Changed)，就會觸發mySliderAction方法。



4.回到程式碼，在ViewController類別裡的mySliderAction方法中，寫入下面的程式碼：

```
@IBAction func mySliderAction(_ sender: UISlider) {  
    print(sender.value)      //印出目前滑桿的數值  
}
```

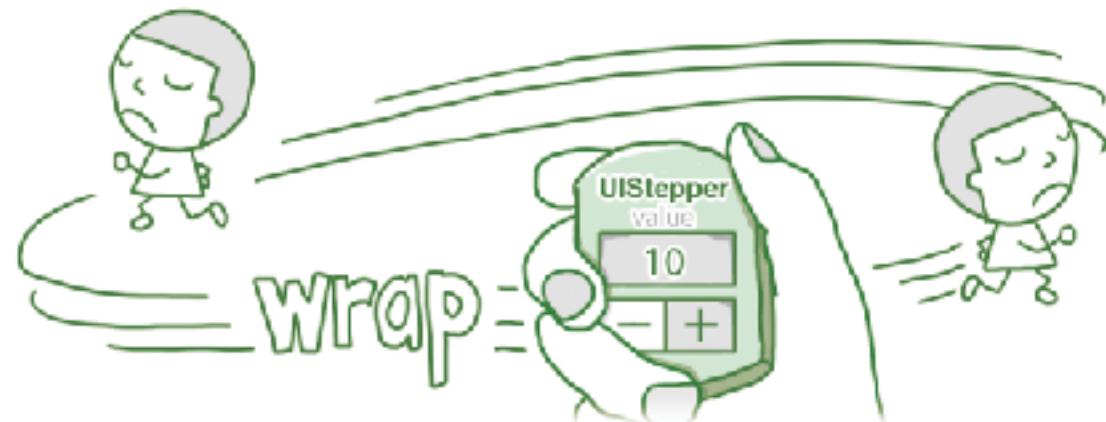
5. 程式碼中，印出滑桿的屬性value、此為目前滑桿的滑動值。不過該數值是有小數點的浮點數。如果要把這樣的浮點數四捨五入變成整數的話，請配合使用整數的初始化函式。經過此函式處理以後，滑桿的數值就會變成整數了：

```
@IBAction func mySliderAction(_ sender: UISlider) {  
    print(Int(sender.value)) //把滑桿的數值轉換成整數  
}
```

【使用程式碼製作滑桿】

範例程式碼同時記錄了如何直接用程式碼生成滑桿(UISlider)。如果要使用程式碼直接生成滑桿的話，請參考本單元的範例程式碼。

使用計數器(UIStepper)



【問題】如何使用計數器

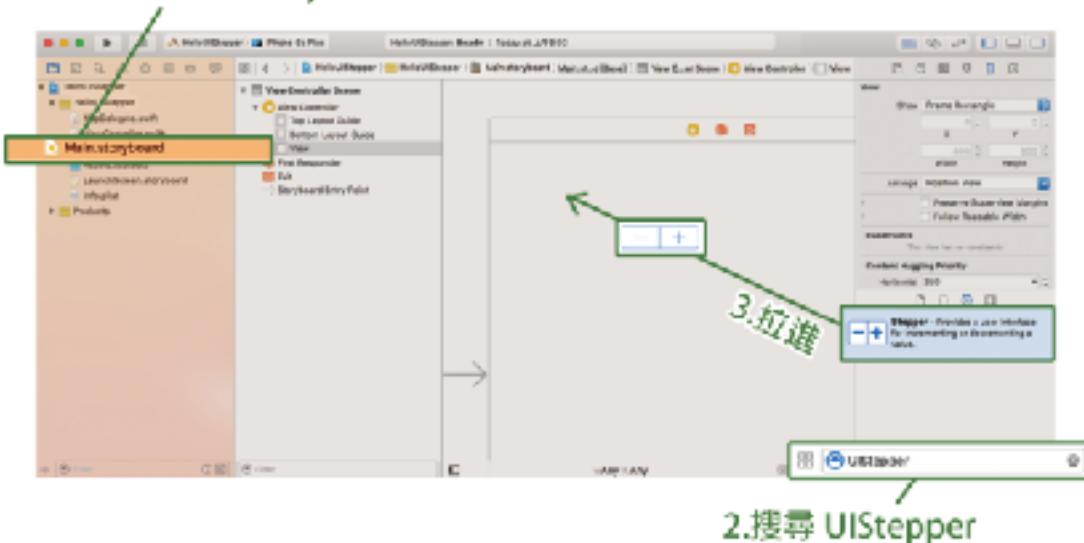
【解答】設定UIStepper，從其屬性value得到目前計數器的值

【範例程式碼】HelloUIStepper

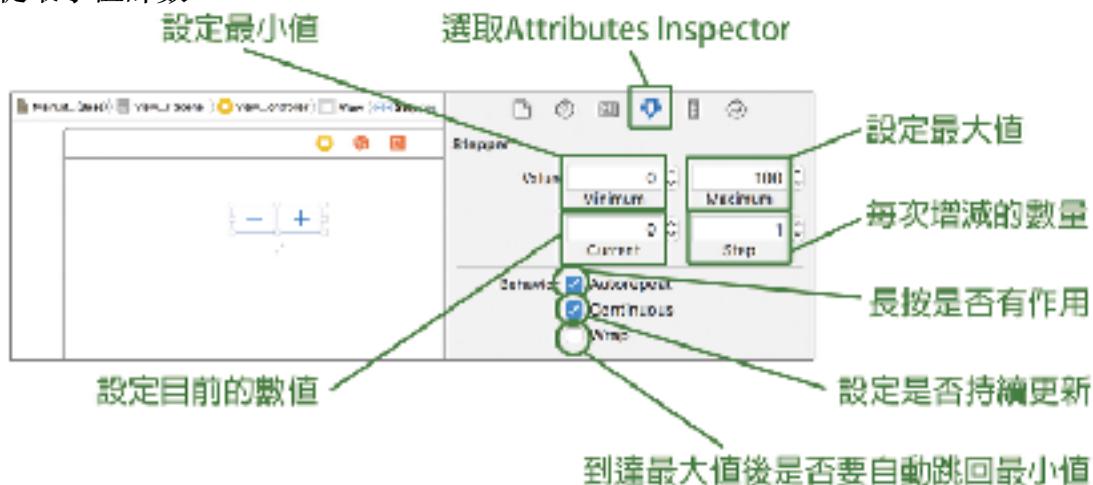
【過程說明】

1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UIStepper、用滑鼠把計數器拉到畫面上。

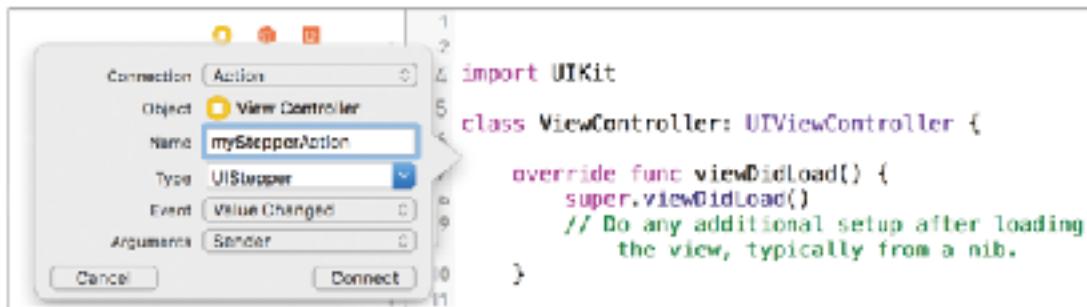
1. 選 Main.storyboard



2. 點選步驟1加入的計數器(UIStepper)，可以在右邊欄設定計數器的最小值、最大值，與目前的數值。預設最小值是0、最大值是100，預設數值為0。除了這些以外，還可以在Step欄裡設定每次增加或是減少的數量。右下方的Autorepeat選項決定長按是否持續會有作用、Continuous選項決定是否在按壓過程中要持續地更新、wrap選項決定到達最大值後是否要自動跳回最小值、再從最小值計數。



3. 如果想要接受使用者觸發的計數器事件，請將UIStepper連結到程式碼中：選擇UIStepper，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIStepper到程式碼中。跳出的對話框裡，請先選擇Action，接著幫這個計數器會觸發的方法命名成myStepperAction。Type一欄請選UIStepper，event選Value Changed。設定完成後，使用者按壓計數器的加號或減號，讓計數器狀態改變時(Value Changed)，就會觸發myStepperAction方法。



4.回到程式碼，在ViewController類別裡的myStepperAction方法中，寫入下面的程式碼：

```
@IBAction func myStepperAction(_ sender: UIStepper) {  
    print(sender.value) //印出目前計數器的數值  
}
```

5.程式碼中，印出計數器的屬性value、此為計數器目前記錄的值。不過該數值是有小數點的浮點數。如果要把這樣的浮點數變成整數的話，請配合使用Int函式。經過此函式處理以後，滑桿的數值就會變成整數了：

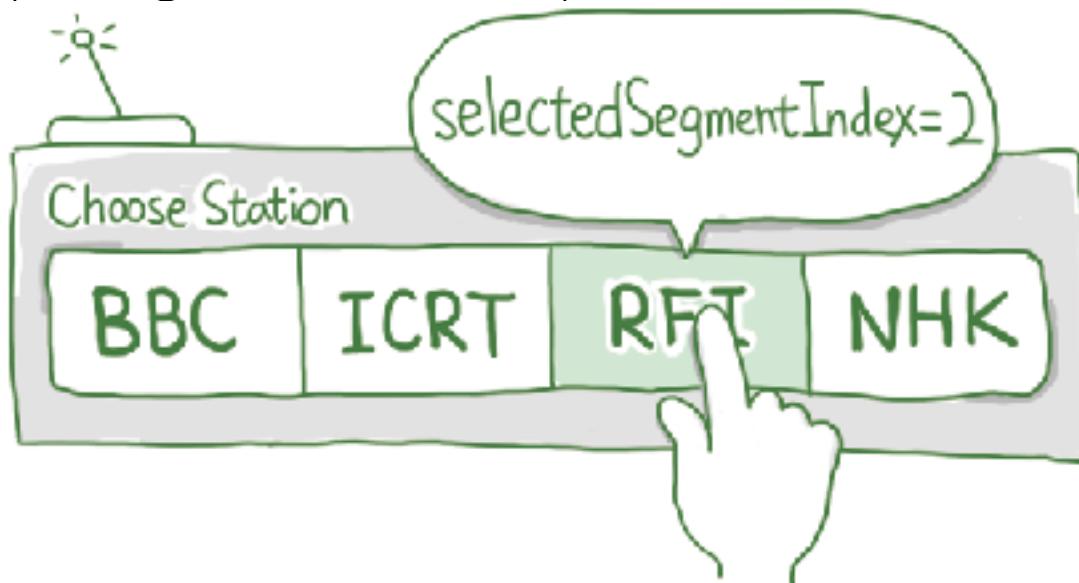
```
@IBAction func myStepperAction(_ sender: UIStepper) {  
    print(Int(sender.value)) //把計數器的值轉換成整數  
}
```

【使用程式碼製作計數器】

範例程式碼同時記錄了如何直接用程式碼生成計數器(UIStepper)。如果要使用程式碼直接生成計數器的話，請參考本單元的範例程式碼。

使用分段式按鈕呈現多個選項

(UISegmentControl)



[問題]如何使用分段式按鈕(UISegmentedControl)

[解答]連結Storyboard上的分段式按鈕到程式碼中，做後續的設定

[範例程式碼]HelloUISegmentedControl

[過程說明]

- 1.請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UISegmentedControl、用滑鼠把此元件拉到畫面上。

1.選 Main.storyboard



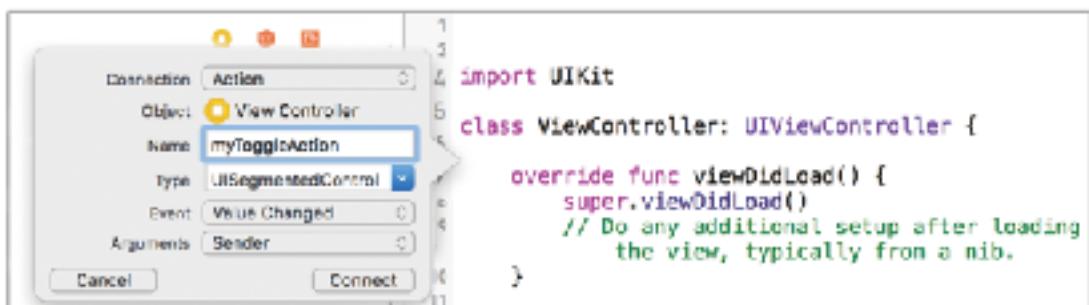
2.搜尋 UISegmentedControl

- 2.點選步驟1加入的UISegmentedControl，可以在右邊欄設定選項的數目。當 UISegmentedControl的長度不夠容納多選項的時候，可以直接拖拉增加寬度。每個分段選項內容可以直接雙擊改變內容，或是在右邊欄的Segment選項先選擇某個段落，再在Title欄改動該段落的顯示內容。

範例中假想在某個社交應用程式中，此UISegmentedControl控制貼文的隱私。
把第一個欄位改成Public，第二個欄位改成Private。



3.如果想要接受使用者到底選擇了那個選項，請將UISegmentedControl連結到
程式碼中：選擇UISegmentedControl，接著選取右上方的Assistant Editor。看到
畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結
UISegmentedControl到程式碼中。跳出的對話框裡，請先選擇Action，接著幫
這個分段式按鈕會觸發的方法命名成myToggleAction。Type一欄請選
UISegmentedControl，event選Value Changed。
設定完成後，使用者按壓分段式按鈕，讓選項改變時(Value Changed)，就會觸
發myToggleAction方法。



4.回到程式碼，在ViewController類別裡的myToggleAction方法中，寫入下面的
程式碼：

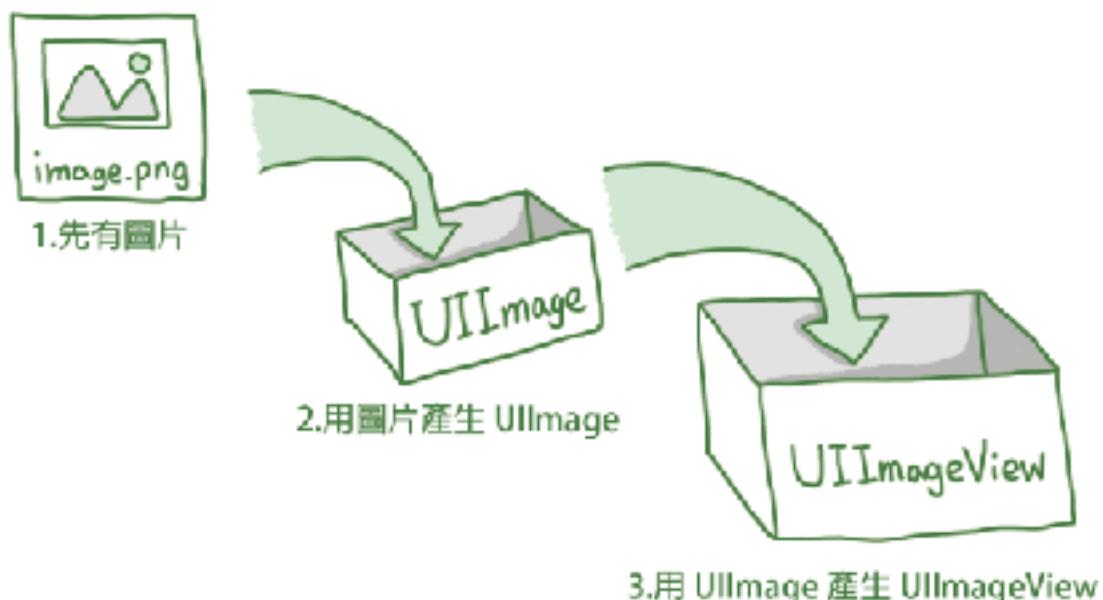
```
@IBAction func myToggleAction(_ sender: UISegmentedControl) {
    if sender.selectedSegmentIndex == 0{
        print("public")           //遇到 index 是 0 的情況
    }else{
        print("private")         //遇到 index 是 1 的情況
    }
}
```

5. 程式碼中，判斷選項的編號(selectedSegmentIndex)，如果選項的編號為0，代表選到的是第一個選項；因為範例中只有兩個選項，所以如果不是第一個選項，就是第二個選項。

[使用程式碼製作分段式按鈕]

範例程式碼同時記錄了如何直接用程式碼生成分段式按鈕(**UISegmentedControl**)。如果要使用程式碼直接生成分段式按鈕的話，請參考本單元的範例程式碼。

顯示圖片(**UIImageView**)



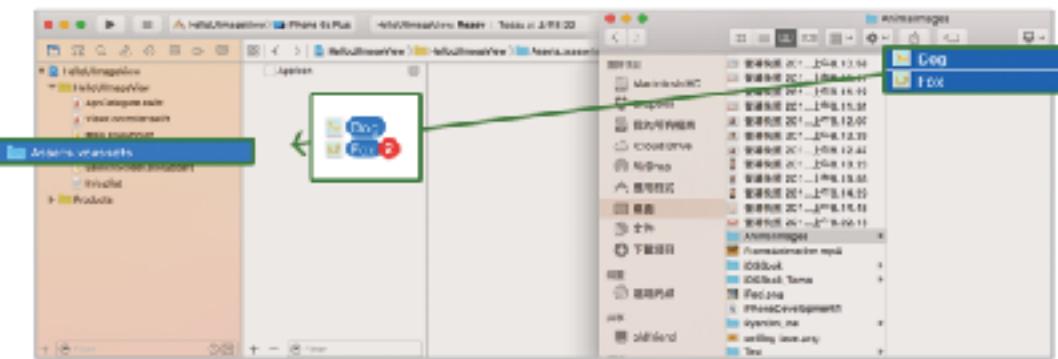
[問題]如何顯示圖片

[解答]把圖片放進專案中，用圖片產生**UIImage**、用**UIImage**產生**UIImageView**，把**UIImageView**放到畫面上，就可以在手機上面顯示圖片

[範例程式碼]**HelloUIImageView**

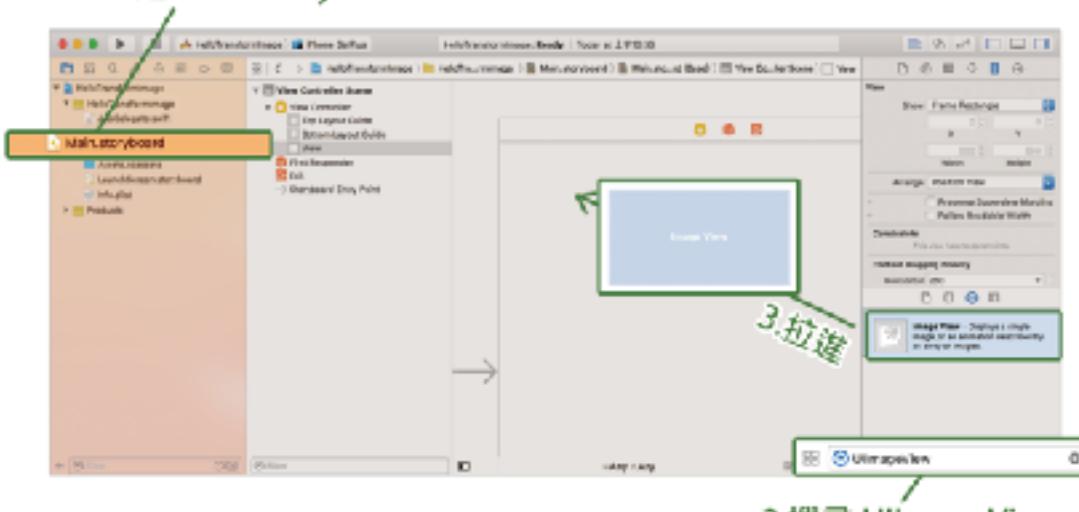
[過程說明]

1. 請先新增一個Single View Application。新增之後，在左邊欄選擇**Assets.xcassets**，把範例中要顯示的圖片匯入到專案中。這兩張圖片分別是狗和狐狸的塗鴉。名稱分別為「Dog.png」與「Fox.png」。這兩張圖片都是正方形，長寬都是612像素。(範例圖片在本章隨書程式碼的**AnimalImages**資料夾中)



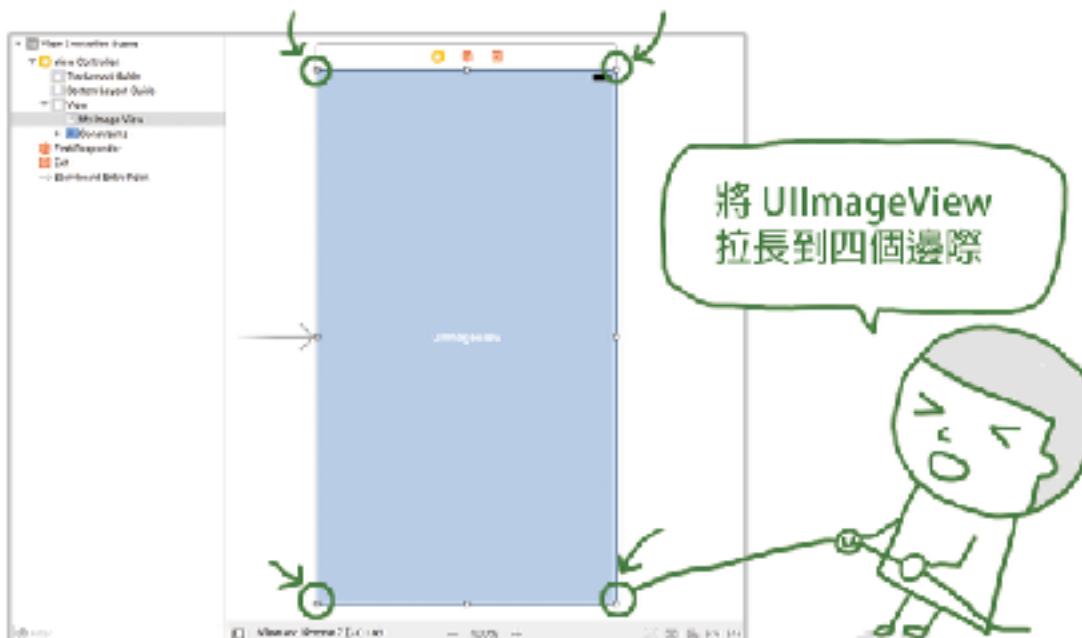
2. 在左邊欄點選Main.storyboard。然後在右下方搜尋UIImageView、用滑鼠把此元件拉到畫面上。

1. 選 Main.storyboard



2. 搜尋 UIImageView

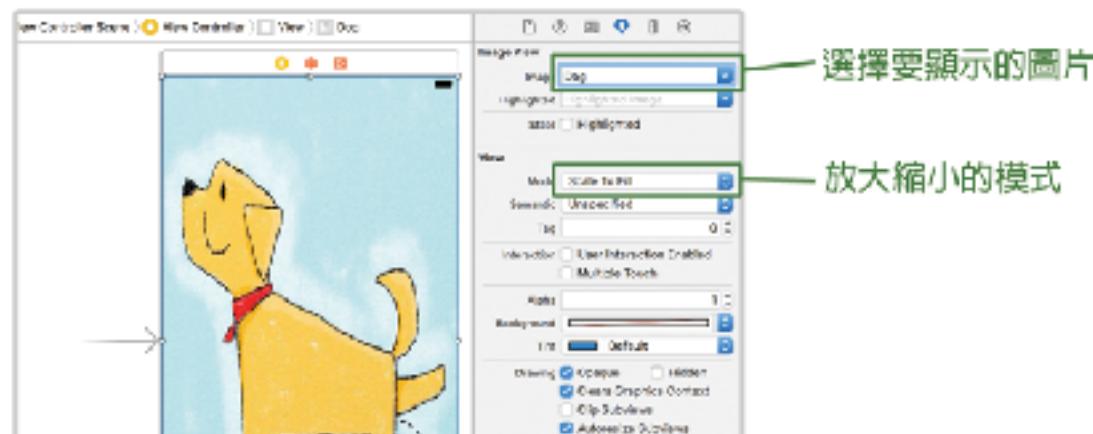
3. 點選步驟2加入的UIImageView，用滑鼠將其四個端點分別拉到四個邊際。



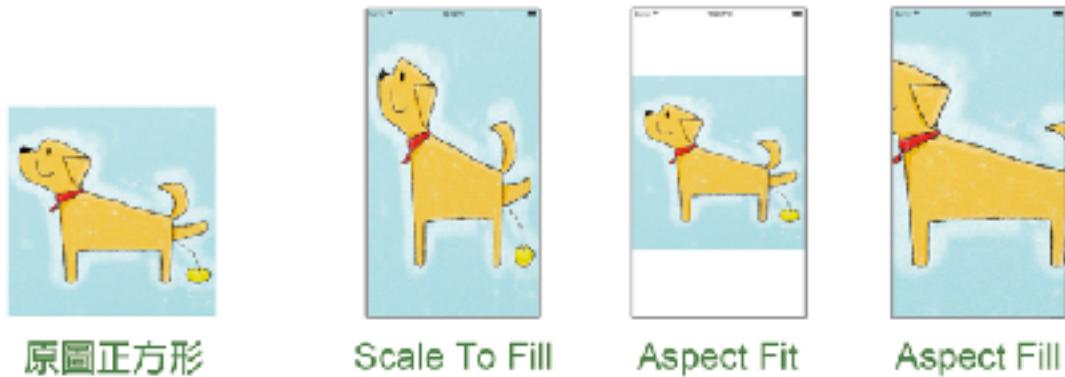
4. 選擇下方的Pin按鈕，設定UIImageView的大小。請如下圖所示，讓用來顯示圖片的UIImageView距離四個邊際的距離都是0。這樣設定的話，執行程式時，用來顯示圖片的UIImageView就會填滿整個螢幕。



5. 設定UIImageView要顯示的圖片：在image欄選Dog，就會顯示之前匯入的、檔名為「Dog.png」的圖片。除此以外，在右邊欄還有設定放大縮小模式的選項。預設是「Scale To Fill」，請改選擇為「Aspect Fit」。



6. UIImageView有很多放大縮小的模式可供選擇。其中最重要的三個如下所示：



a) Scale To Fill：不等比例地放大縮小。讓圖片填滿整個UIImageView。

範例中，原圖是正方形、顯示圖片的UIImageView設定成跟手機的長方形畫面一樣大。顯示圖片選Scale To Fill，就會不等比例的放大縮小填滿整個UIImageView，圖片會有拉長的感覺。

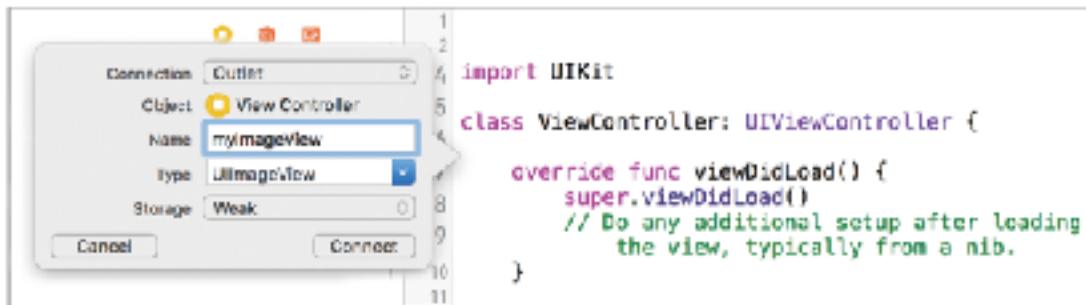
b) Aspect Fit：等比例地放大縮小，到其中的一個邊際再放大就要超過UIImageView的範圍時停止放大。

範例中，選Aspect Fit放大到寬度與UIImageView同寬時就停止放大縮小。

c) Aspect Fill：等比例地放大縮小。其中的一個邊際再放大就要超過UIImageView的範圍時還不停止放大，直到全部的邊際再放大就要超過UIImageView的範圍時才停止放大縮小。。

範例中，選Aspect Fill放大，結果會在圖片的高度與UIImageView的高度同高時才停止放大縮小。

7. 目前顯示的圖片是小狗的圖片「Dog.png」。如果想要用程式碼更改UIImageView顯示的圖片，請將UIImageView連結到程式碼中：選擇UIImageView，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIImageView到程式碼中。跳出的對話框裡，把UIImageView命名成myImageView。之後在程式碼裡，屬性myImageView就代表畫面上的UIImageView。



4.回到程式碼，在ViewController類別裡的viewDidLoad方法中，寫入下面的程式碼：

```
override func viewDidLoad() {  
    super.viewDidLoad()  
    let foxImage = UIImage(named: "Fox")      // 讀入狐狸圖片  
    myImageView.image = foxImage                // 設定顯示狐狸圖片的UIImage  
}
```

5.程式碼中，先用狐狸圖片建立了UIImage型別的物件foxImage。接下來把foxImage設定給myImageView的image屬性。畫面上的圖片就會從小狗的「Dog.png」改變成「Fox.png」。

【幫UIImageView加上外框線】

請在ViewController類別中的viewDidLoad方法中，再加入下面的程式碼：

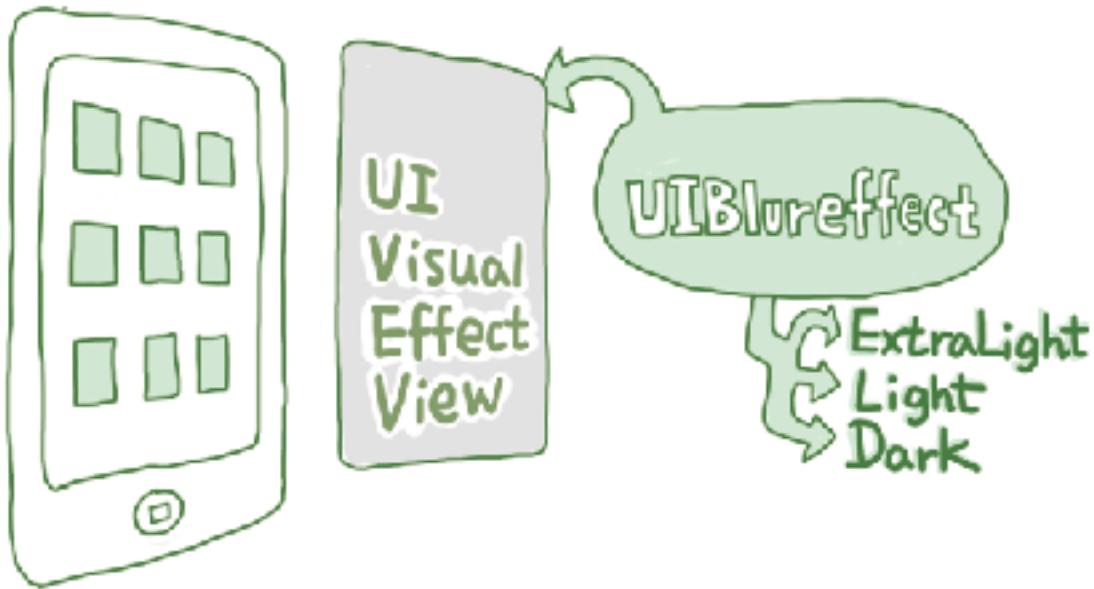
```
// 設定CALayer的邊線顏色  
myImageView.layer.borderColor = UIColor.black.cgColor  
// 設定邊線粗細  
myImageView.layer.borderWidth = 2.0
```

幫UIImageView加上外框線的方法很簡單，只要設定其CALayer的邊線顏色與邊線粗細，就可以幫UIImageView加上外框線。

【使用程式碼製作UIImageView來顯示圖片】

範例程式碼同時記錄了如何直接用程式碼生成UIImageView。如果要使用程式碼直接生成UIImageView來顯示圖片或照片的話，請參考本單元的範例程式碼。

做出模糊的效果



[問題]如何做出模糊的效果

[解答]選擇模糊的效果產生 `UIBlurEffect` 物件，再用 `UIBlurEffect` 產生 `UIVisualEffectView` 物件。把 `UIVisualEffectView` 加入畫面中，就能做出模糊效果。

[範例程式碼] HelloUIVisualEffectView

[過程說明]

1. 請先依照上一個[顯示圖片]問題解答的前5個步驟建立新專案，並且在畫面上新增一個 `UIImageView`，讓畫面上顯示一張圖片。
2. 接下來在 `ViewController` 類別的 `viewDidLoad` 方法中，用程式碼建立 `UIVisualEffectView` 讓圖片模糊：

```
override func viewDidLoad() {
    super.viewDidLoad()
    let blurEffect = UIBlurEffect(style: .light)           // 選擇模糊效果
    let blurView = UIVisualEffectView(effect: blurEffect)// 建立模糊視圖
    blurView.frame = view.frame                           // 設定模糊視圖大小
    blurView.center = view.center                         // 設定模糊視圖位置
    view.addSubview(blurView)                            // 把模糊視圖加入畫面
}
```

3.1. 建立 `UIVisualEffectView` 的方法很直覺。先選擇要建立的模糊效果，把選項存在 `UIBlurEffect` 物件中。有三種選項：

- a) dark(暗色模糊)
- b) light(淡色模糊)
- c) extraLight(亮色模糊)
- d) prominent(突出：這種效果只有 iOS 10 以後才支援)
- e) regular(正常：這種效果只有 iOS 10 以後才支援)

3.2 然後用 `UIBlurEffect` 物件建立 `UIVisualEffectView`，調整該視圖的大小及位置。這個 `UIVisualEffectView` 就好像是一片毛玻璃，加入畫面之後，會讓下層的

所有圖像都模糊了起來。

製作圖像效果：圓角圖像、圓形圖像、旋轉、位移，與放大縮小圖像



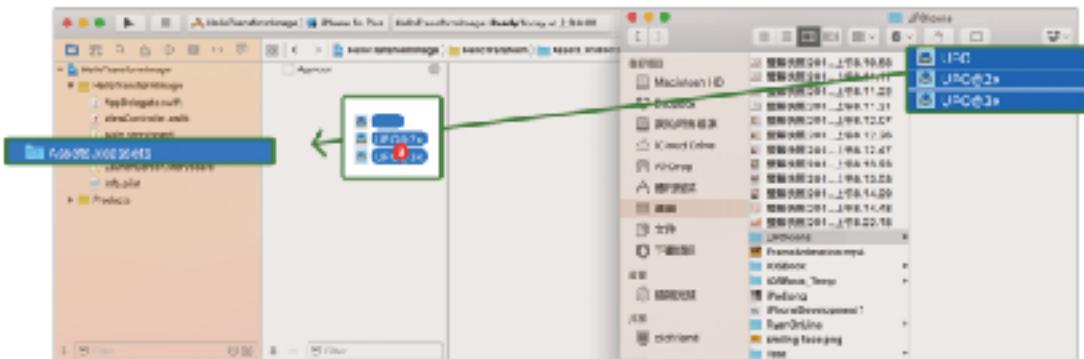
[問題]如何幫圖片做出圓角效果

[解答]設定CALayer的cornerRadius屬性可以設定圓角；設定視圖的transform可以旋轉、位移與放大縮小圖像。

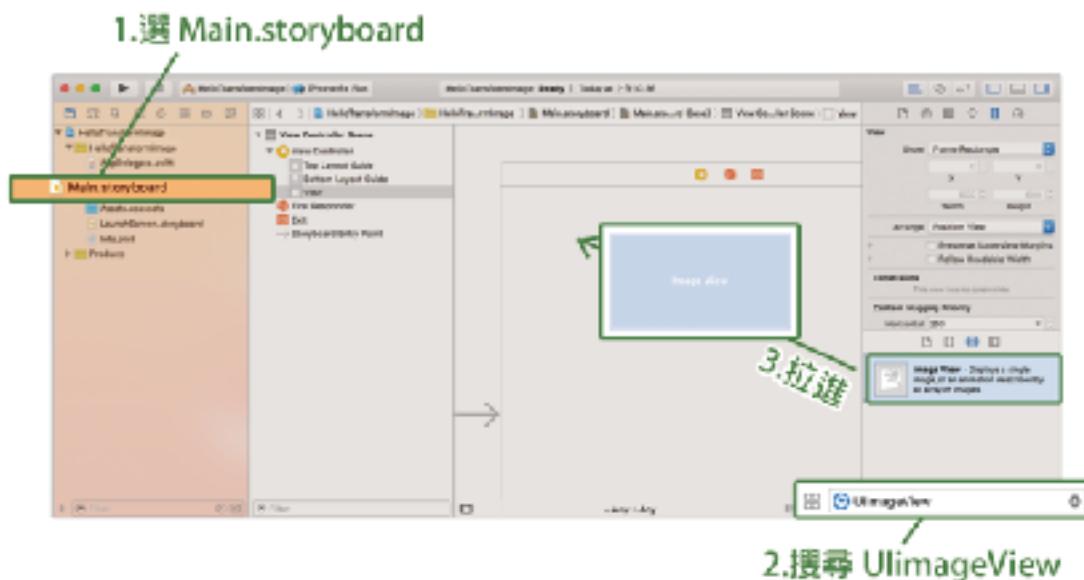
[範例程式碼]HelloTransformImage

[圖像設定]

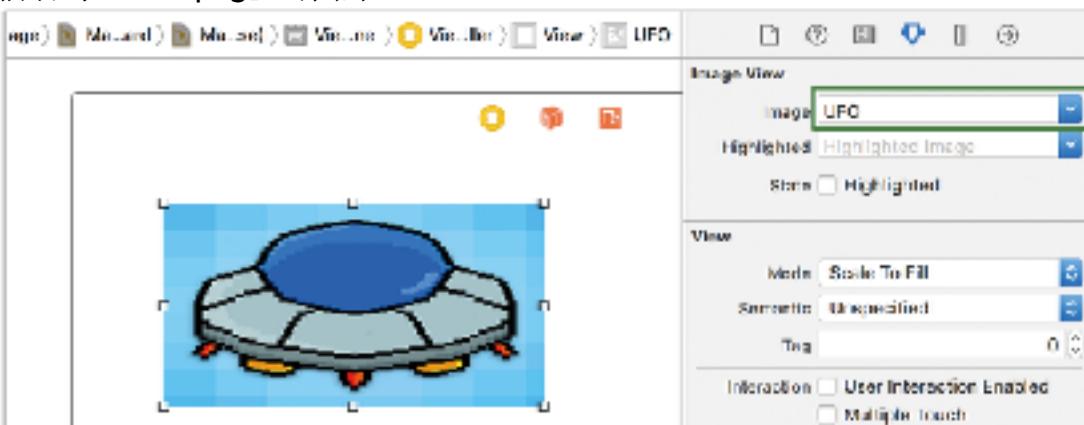
1. 請先新增一個Single View Application。新增之後，在左邊欄選擇Assets.xcassets，把範例中要顯示的圖片匯入到專案中。匯入的圖片是正方形的飛碟卡通圖片，名稱為「UFO.png」。(範例圖片在本章隨書程式碼的Ufolcons資料夾中)



2. 在左邊欄點選Main.storyboard。然後在右下方搜尋UIImageView、用滑鼠把此元件拉到畫面上。



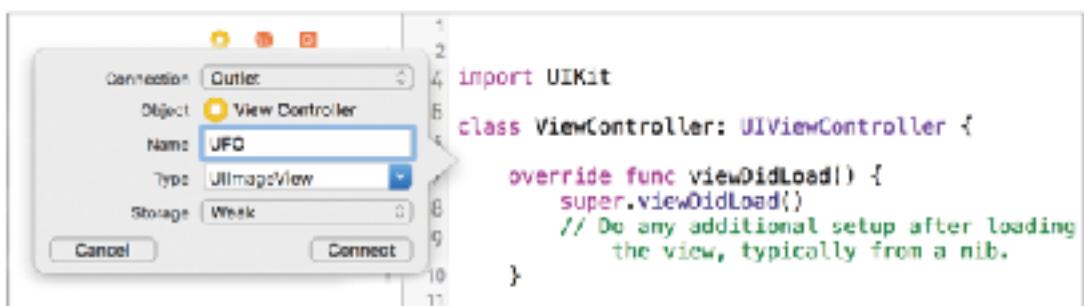
3. 設定UIImageView要顯示的圖片：在image欄選UFO，就會顯示之前匯入的、檔名為「UFO.png」的圖片。



4. 除此以外，請選擇Size Inspector，設定這張圖的座標、寬度與高度。範例中把圖片的x座標設在200、y座標設在200，寬度200，高度200。



5.如果要用程式碼將圖片改成圓角，請將飛碟圖片連結到程式碼中：選擇UFO圖片，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結飛碟圖片到程式碼中。跳出的對話框裡，把UIImageView命名成UFO。之後在程式碼裡，屬性UFO就代表畫面上的飛碟圖片。



【幫圖片加上圓角】

請在ViewController類別裡的viewDidLoad方法中，寫入下面的程式碼：

```
override func viewDidLoad() {
    super.viewDidLoad()
    //圓角
    UFO.layer.cornerRadius = 40
    UFO.clipsToBounds = true
}
```

設定視圖CALayer的cornerRadius屬性，此值越大，圓角越圓。再設定視圖的clipsToBounds屬性為true，圖片就會出現圓角的效果。

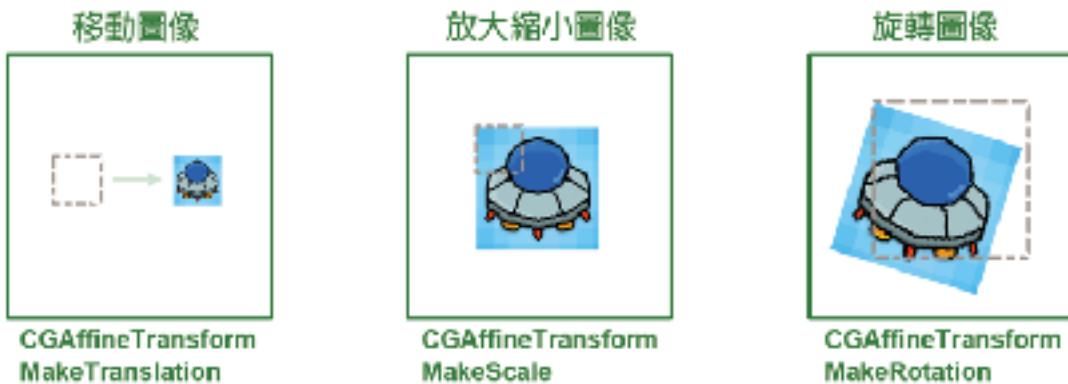
【把圖片變成圓形】

改動原來在ViewController類別裡的viewDidLoad方法，把圓角的兩行程式碼刪掉，代換成下面的程式碼：

```
//圓形  
UFO.layer.cornerRadius = 100  
UFO.clipsToBounds = true
```

視圖CALayer的cornerRadius值越大，圓角越圓。如果在圖像為正方形的情況下、此值大到圖寬度的一半，圖片就會變成圓形。

【移動圖片】



接下來要介紹三種簡單的變形方法，第一種是移動圖片。請改動原來在ViewController類別裡的viewDidLoad方法，代換成下面的程式碼：

```
//移動  
UFO.transform = CGAffineTransform(translationX: -200, y: -200)
```

呼叫CGAffineTransform(translationX:, y:)方法就可以設定移動變形。其中第一個參數指定x方向的位移；第二個參數指定y方向的位移。把這樣的變形指定給視圖的transform屬性，圖像就會移動了。範例中，x跟y方向的位移都設定成-200，於是飛碟以原位為基準，往左邊移動200，往上移動200。

【旋轉圖片】

如果要旋轉圖片的話，請刪除之前撰寫的程式碼。改成下面的程式碼：

```
//旋轉  
UFO.transform =  
    CGAffineTransform(rotationAngle: CGFloat(30.0 * M_PI / 180.0))
```

呼叫`CGAffineTransform(rotationAngle:)`方法就可以設定旋轉變形。其中的參數是旋轉的度數。把這樣的變形指定給視圖的`transform`屬性，圖像就會旋轉了。範例中把30的角度轉換成度數當作參數呼叫該變形方法，飛碟圖片就會旋轉30度。

【放大縮小圖片】

如果要放大縮小圖片的話，請再次刪除之前撰寫的程式碼，代換成下面的程式碼：

```
//放大縮小
UFO.transform = CGAffineTransform(scaleX: 0.5, y: 0.5)
```

呼叫`CGAffineTransform(scaleX: y:)`方法就可以設定縮放變形。其中第一個參數指定x方向放大縮小的比例；第二個參數指定y方向放大縮小的比例。把這樣的變形指定給視圖的`transform`屬性，圖像就會放大縮小了。範例中，x跟y方向放大縮小的比例都設定成0.5，於是飛碟會縮小成原大小的一半。

【結合兩種變形】

上面介紹了移動、旋轉與放大縮小三種將視圖變形的方法，如果要結合兩種變形方法的話，請將原來的程式碼，代換成下面的樣子：

```
//結合兩種
let myTransform1 = CGAffineTransform(scaleX: 0.5, y: 0.5)
let myTransform2 =
    CGAffineTransform(rotationAngle: CGFloat(30.0 * M_PI / 180.0))
UFO.transform = myTransform1.concatenating(myTransform2)
```

呼叫變形設定的`concatenating`方法就可以結合兩種變形。先讓第一個變形的設定，呼叫其`concatenating`方法，參數帶入第二個變形設定，就可以結合兩種變形的結果。範例中，先做出第一個縮小的變形，再做出第二個旋轉的變形，把兩個變形結合起來，圖像就會又縮小又旋轉。

【只要是UIView都可以變形與加圓角】

本範例使用`UIImageView`來做各種變形。事實上，所有`UIView`的子類別都可以設定圓角、放大縮小、旋轉以及移動的變形。請參考範例程式碼，裡面的註釋記錄了以`UIView`為範例變形的作法。

顯示頁數的UIPageControl



[問題]如何UIPageControl來顯示目前的頁數

[解答]使用UIPageControl的currentPage屬性，就可以顯示目前的頁數

[範例程式碼]HelloUIPageControl

[過程說明]

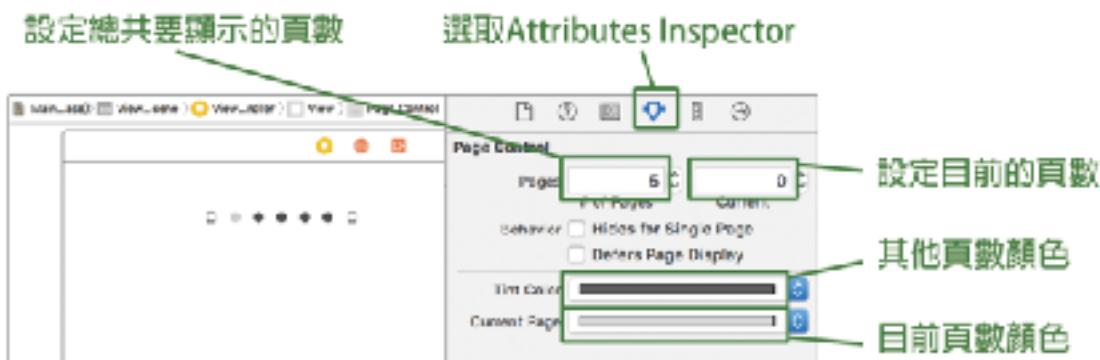
- 1.請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UIPageControl、用滑鼠把UIPageControl拉到畫面上。

1.選 Main.storyboard

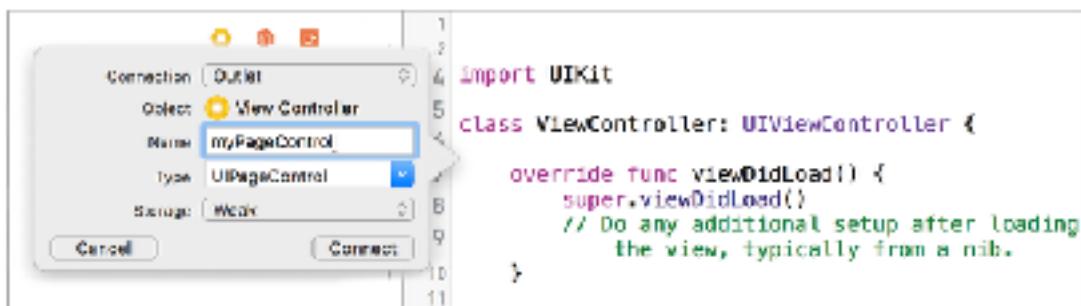


2.搜尋 UIPageControl

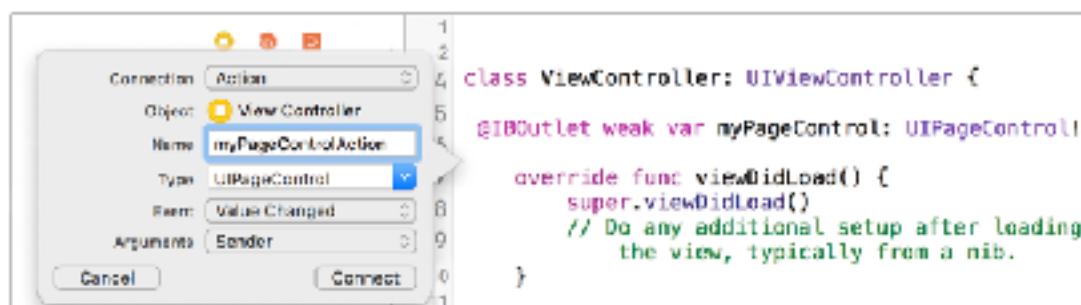
- 2.點選步驟1加入的UIPageControl，可以在右邊欄做出各種相關設定。範例中，將頁數顏色設定成深灰色、目前頁數的顏色設定成淡灰色。這樣設定之下，才能在一片底色為白色的畫面上，看到UIPageControl。



3.如果要從程式碼控制UIPageControl選取的頁數，請連結UIPageControl到程式碼中：選擇UIPageControl，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIPageControl到程式碼中。本範例在跳出的對話框中，把UIPageControl命名成myPageControl。



4.重複步驟3的動作，再次連結文字輸入框到程式碼中。只不過這次跳出對話框裡，在Connection選擇Action、幫這個文字輸入框觸發的事件(IBAction)命名成myPageControlAction，並把Type選擇成UIPageControl。連結完成後，使用者和UIPageControl互動後，會觸發myPageControlAction方法。



4.回到程式碼，請如下修改ViewController類別裡的程式碼：

```

class ViewController: UIViewController {
    @IBOutlet weak var myPageControl: UIPageControl!
    @IBAction func myPageControlAction(_ sender: UIPageControl) {
        print(sender.currentPage) //目前選到的頁數
    }
    override func viewDidLoad() {
        super.viewDidLoad()
        myPageControl.currentPage = 1 //設定目前的頁數
    }
}

```

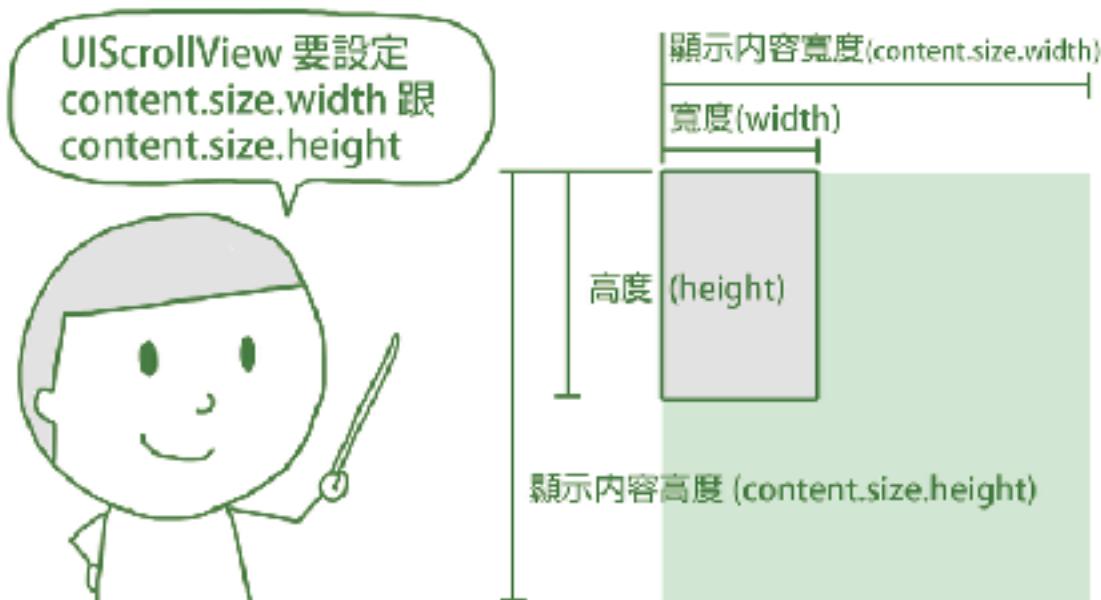
5. 程式碼中，myPageControl代表畫面上的UIPageControl，使用者在和該元件互動時，會觸發myPageControlAction。

範例中只是把目前改變後的頁數印出來。在後面章節介紹過UIScrollView之後，將會進一步介紹如何讓UIPageControl和UIScrollView互動、讓使用者可以利用UIPageControl讓UIScrollView的內容換頁。

[使用程式碼製作UIPageControl]

範例程式碼同時記錄了如何直接用程式碼生成UIPageControl。如果要使用程式碼直接生成該元件的話，請參考本單元的範例程式碼。

使用ScrollView顯示比螢幕大的內容



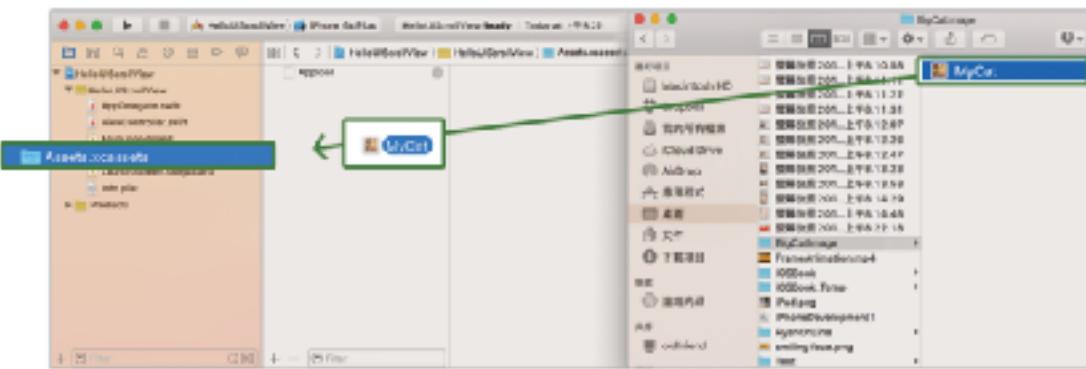
[問題]如何使用UIScrollView顯示比螢幕大的內容

[解答]設定其屬性content.size.width與content.size.height，就可以顯示比其大小還要大的圖片

[範例程式碼]HelloUIScrollView

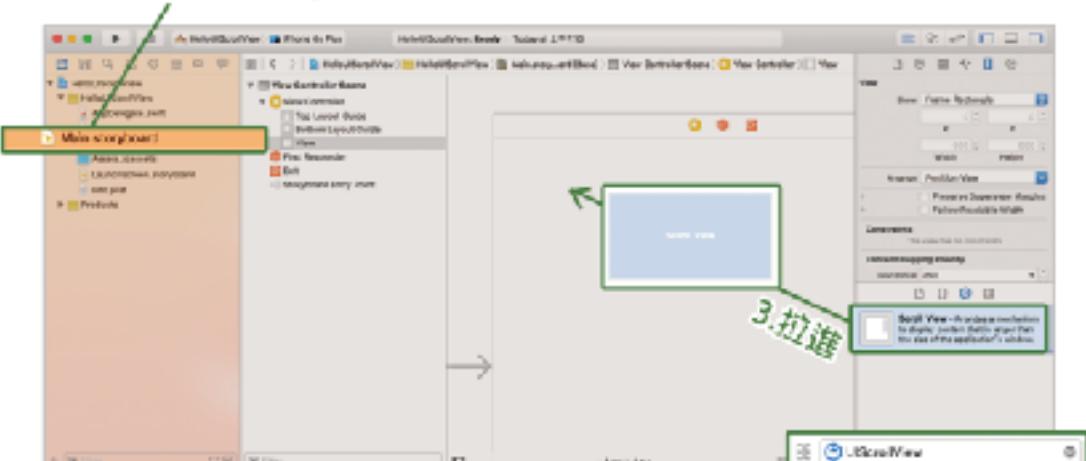
【過程說明】

1. 請先新增一個Single View Application。新增之後，在左邊欄選擇 Assets.xcassets，把範例中要顯示的圖片匯入到專案中。這是一張正方形的照片，名稱為「MyCat.png」，長寬都是1080像素。(範例圖片在本章隨書程式碼的 BigCatImage 資料夾中)



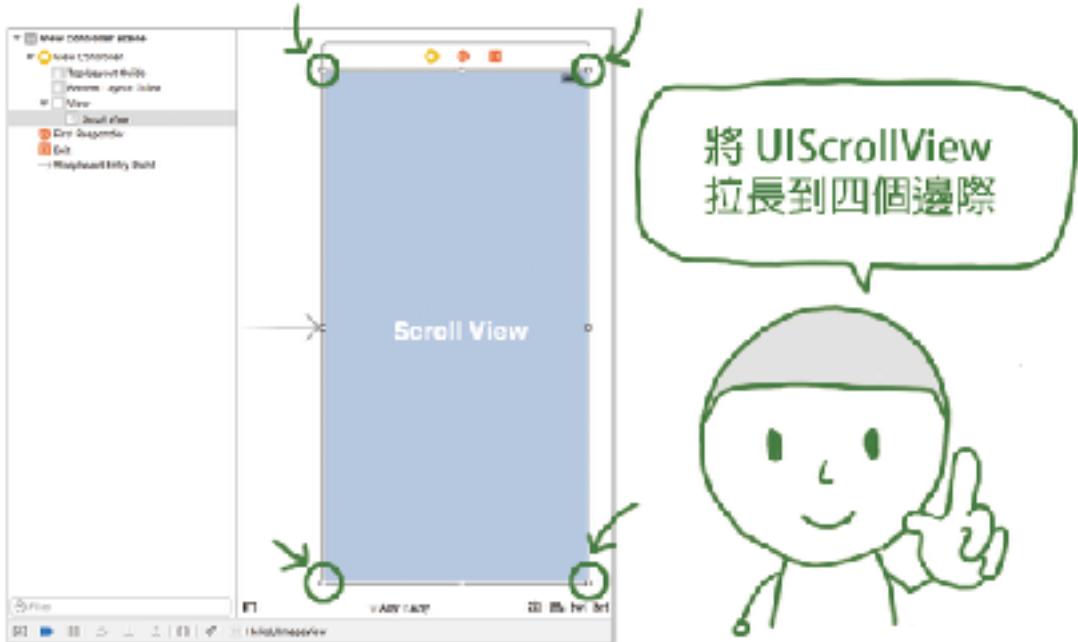
2. 在左邊欄點選Main.storyboard。然後在右下方搜尋UIScrollView、用滑鼠把此元件拉到畫面上。

1. 選 Main.storyboard

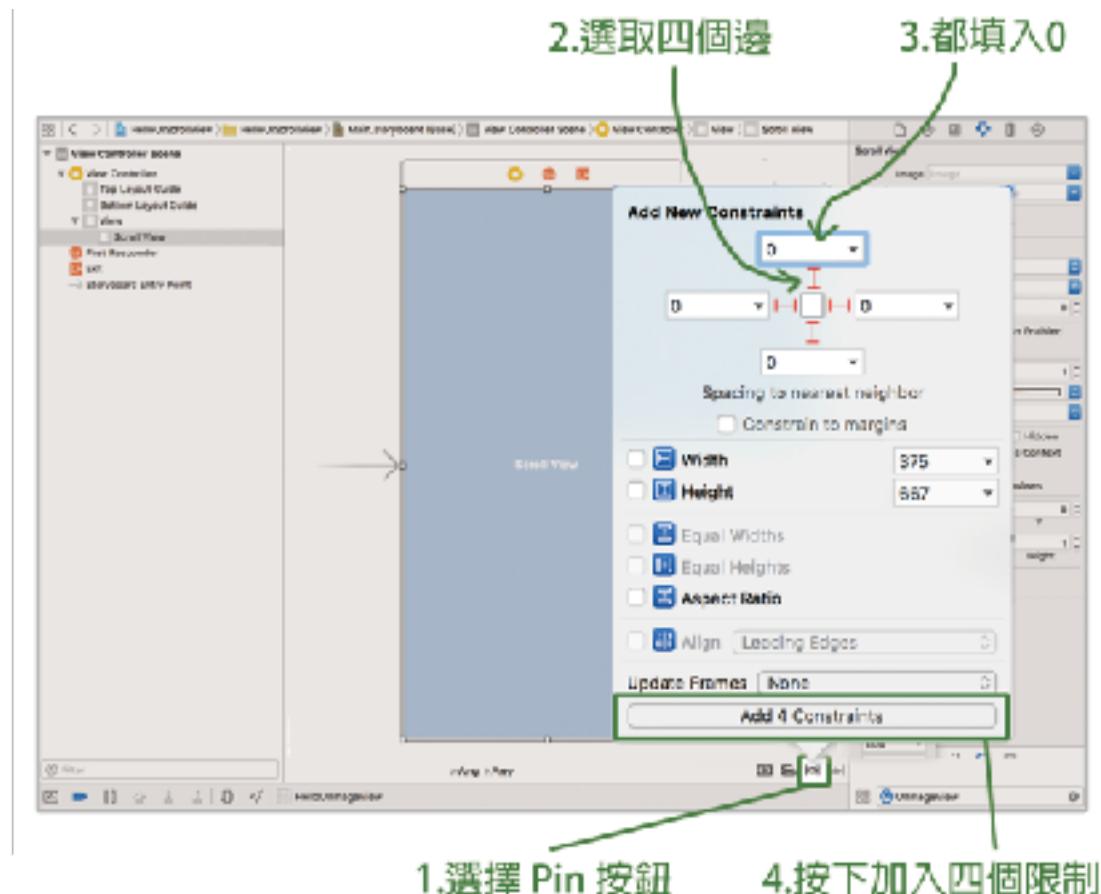


2. 搜尋 UIScrollView

3. 點選步驟2加入的UIScrollView，用滑鼠將其四個端點分別拉到四個邊際。

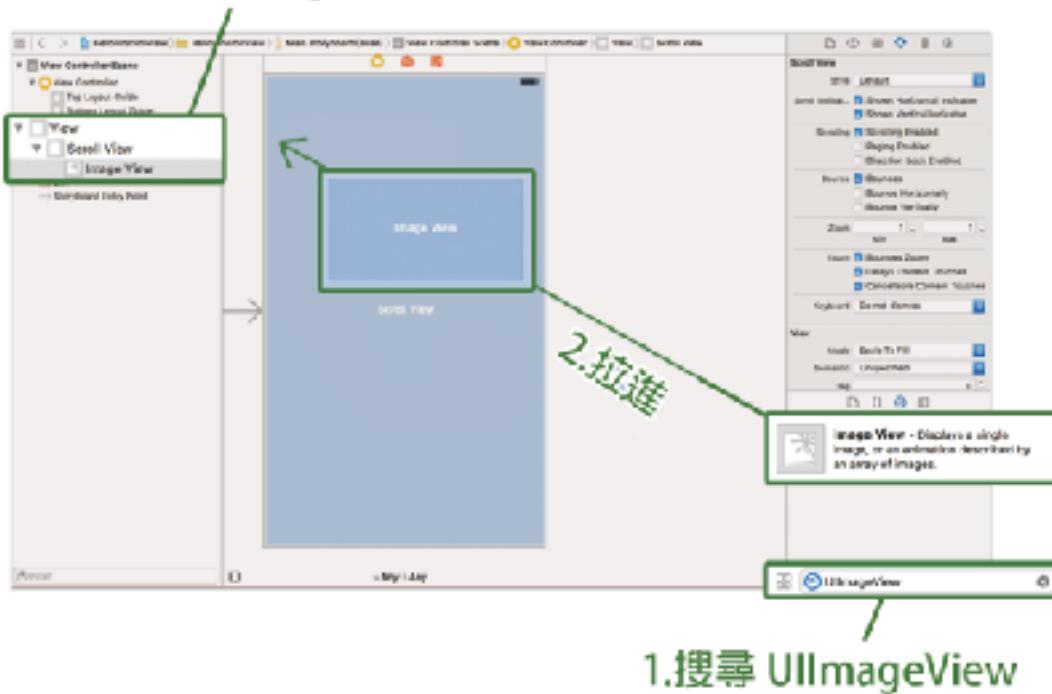


4. 選擇下方的Pin按鈕，設定UIScrollView的大小。請如下圖所示，讓用來顯示圖片的UIScrollView距離四個邊際的距離都是0。這樣設定的話，執行程式時，該元件就會填滿整個螢幕。



5. 在右下方搜尋UIImageView。把UIImageView拉到ScrollView裡面。請注意，拉動結束後，左方欄的層級如圖所示，請確定ImageView在ScrollView裡面。

3. 確定 Image View 在 Scroll View 裡面

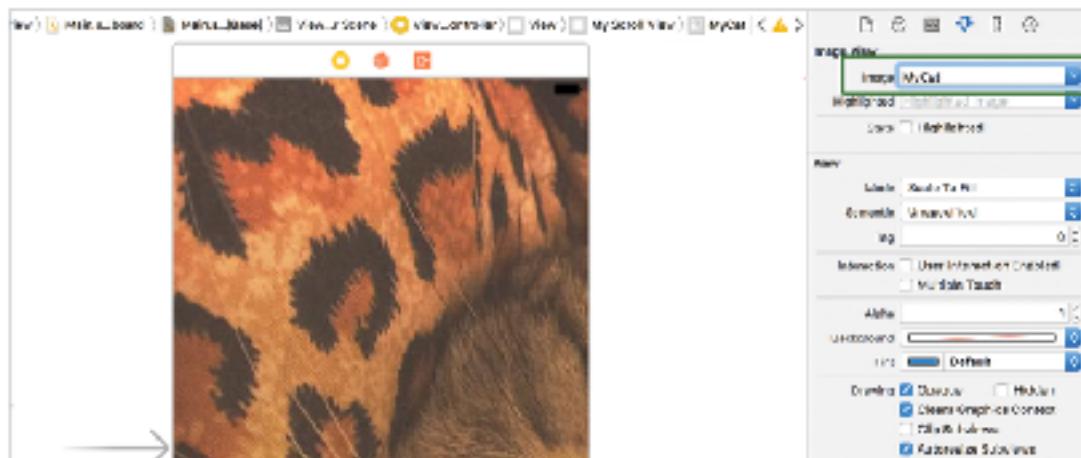


6. 設定UIImageView的座標、寬度跟高度。由於本來圖片的寬度跟高度都是1080。所以設定成1080。x與y座標的部分都設成0。設定完成之後，按下下方的Pin按鈕，固定距離上方與左方的數值。把這兩個數值設定成0，圖片就會緊靠著做左上方。最後固定圖片的寬度與高度為1080。全部設定完成後，按下下方的[Add 4 Constraints]按鈕。

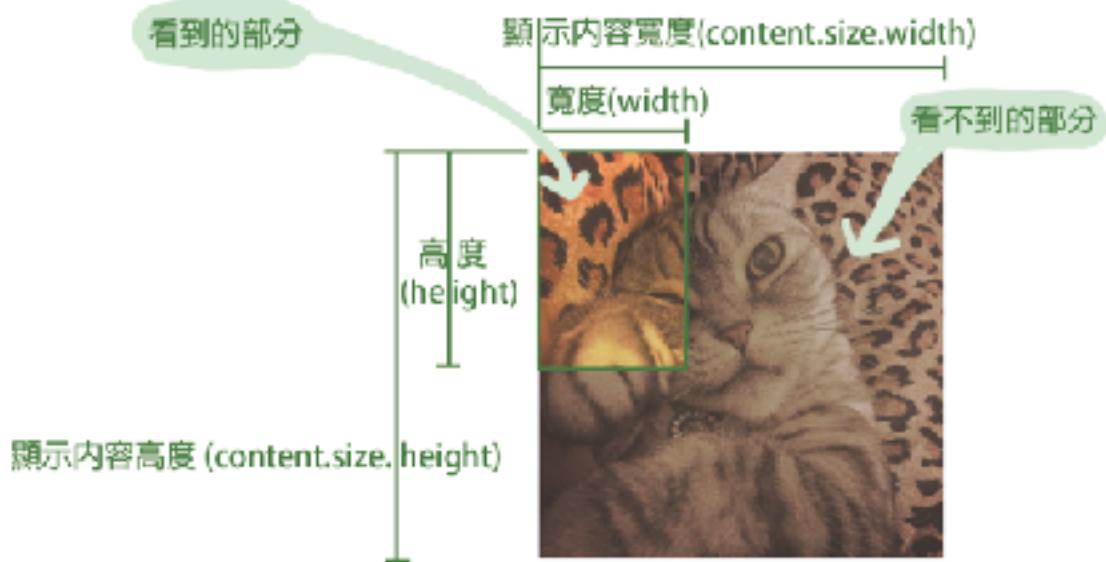
2. 設定座標、寬度跟高度 1. 選 Size Inspector



7. 設定UIImageView要顯示的圖片：在image欄選MyCat，就會顯示之前匯入的、檔名為「MyCat.png」的圖片。

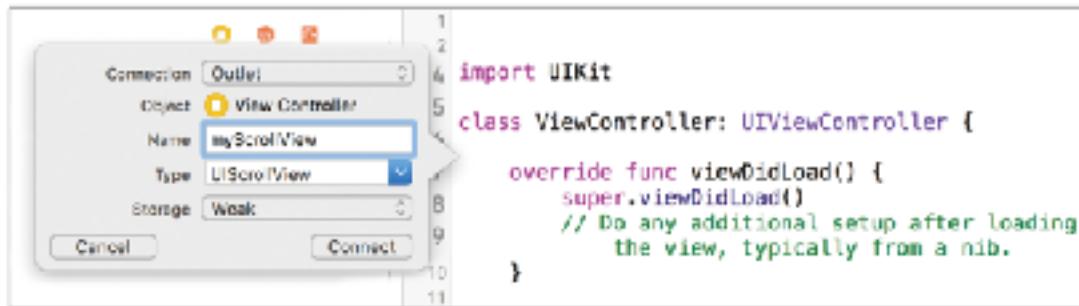


8. 目前做到的進度如下：



目前在畫面加上了ScrollView，讓ScrollView的大小跟手機畫面的大小相同。接著在ScrollView上加入了一張很大的照片。現在執行程式的話，沒有辦法看到圖片其他的部分。需要設定ScrollView的屬性`content.size.width`與`content.size.height`。把這兩個屬性設定成圖片的寬度及高度。UIScrollView就可以以滑動的方式顯示比螢幕大的內容。

9.如果要設定UIScrollView顯示的內容的寬度與高度，需將UIScrollView連結到程式碼中：選擇UIScrollView，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIScrollView到程式碼中。跳出的對話框裡，把UIScrollView命名成myScrollView。之後在程式碼裡，屬性myScrollView就代表畫面上的UIScrollView。



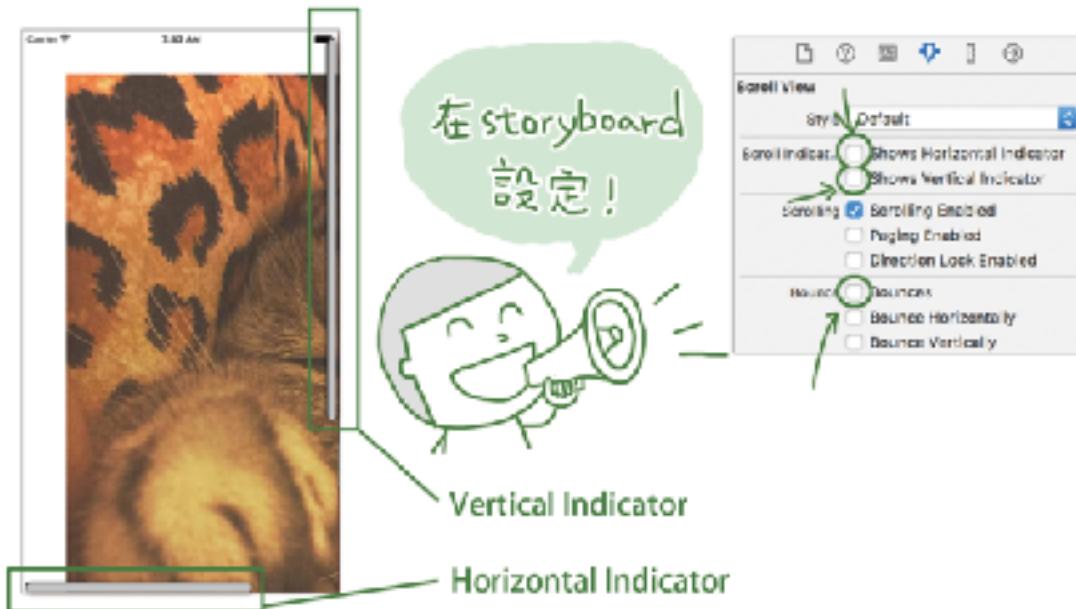
10.回到程式碼，在ViewController類別裡的viewDidLoad方法中，寫入下面的程式碼：

```
override func viewDidLoad() {
    super.viewDidLoad()
    myScrollView.contentSize.width = 1080      // 設定顯示內容寬度
    myScrollView.contentSize.height = 1080      // 設定顯示內容高度
}
```

11.程式碼中，設定了ScrollView顯示內容的寬度與高度，使用者就利用在

ScrollView上滑動手指看到完整的圖片了。

12. 使用ScrollView的過程中，會發現滑動的時候，右邊和下面會出現滑桿。這個叫做Vertical與Horizontal Indicator。如果不想顯示滑桿的話，請到Main.storyboard，取消勾選顯示這兩個滑桿。另外當滑動到邊界時，ScrollView會出現白底和彈跳的效果。如果不想出現這樣的效果，請取消勾選Bounces的選項。



【放大縮小圖片】

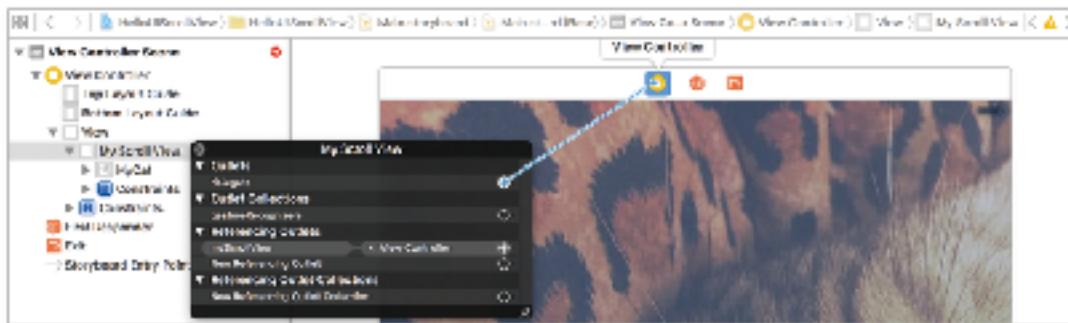


如果想要做出用手指放大縮小照片的功能，要...

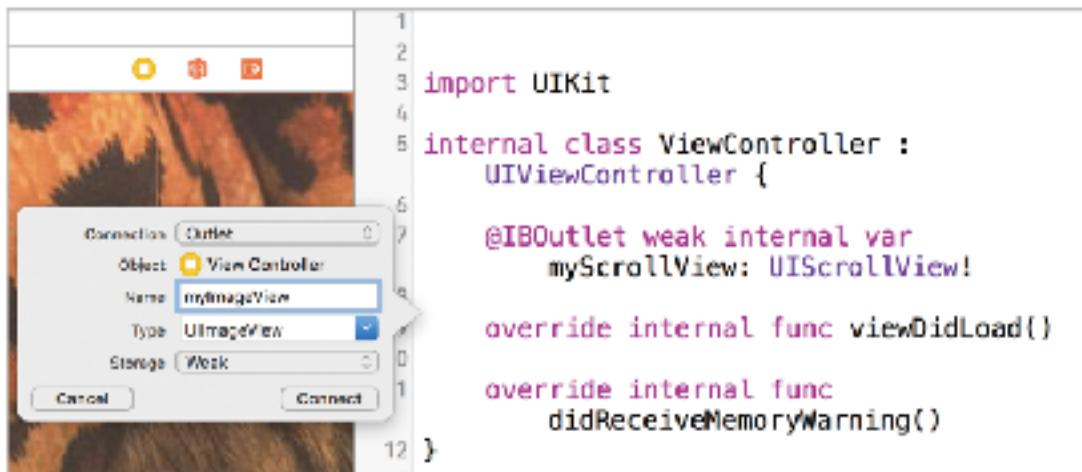
- a.讓ViewController類別服從UIScrollViewDelegate協定(Protocol)。
- b.把ViewController類別當做ScrollView的Delegate。
- c.覆寫UIScrollViewDelegate中的viewForZoomingInScrollView方法。

下面記錄詳細的流程：

- 1.到Main.storyboard，選擇ScrollView。按下右鍵，在跳出的對話框中，選擇delegate。把delegate右邊的圓點連結到ViewController類別。這樣就把順利地把ViewController類別當做ScrollView的Delegate。



- 2.把ImageView連結到程式碼中：選擇ScrollView裡面的UIImageView，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIImageView到程式碼中。跳出的對話框裡，把UIImageView命名成myImageView。之後在程式碼裡，屬性myImageView就代表UIScrollView上的UIImageView。



3.修改ViewController類別裡面的程式碼：

不要忘了加入這個
/

```

class ViewController: UIViewController, UIScrollViewDelegate {
    @IBOutlet weak var myScrollView: UIScrollView!
    @IBOutlet weak var myImageView: UIImageView!
    override func viewDidLoad() {
        super.viewDidLoad()
        myScrollView.contentSize.width = 1080           //設定顯示內容寬度
        myScrollView.contentSize.height = 1080          //設定顯示內容高度
        myScrollView.minimumZoomScale = 1.0            //設定最小縮放比例
        myScrollView.maximumZoomScale = 5.0            //設定最大縮放比例
    }
    func viewForZooming(in scrollView: UIScrollView) -> UIView? {
        return myImageView                         //要放大縮小的圖片
    }
}

```

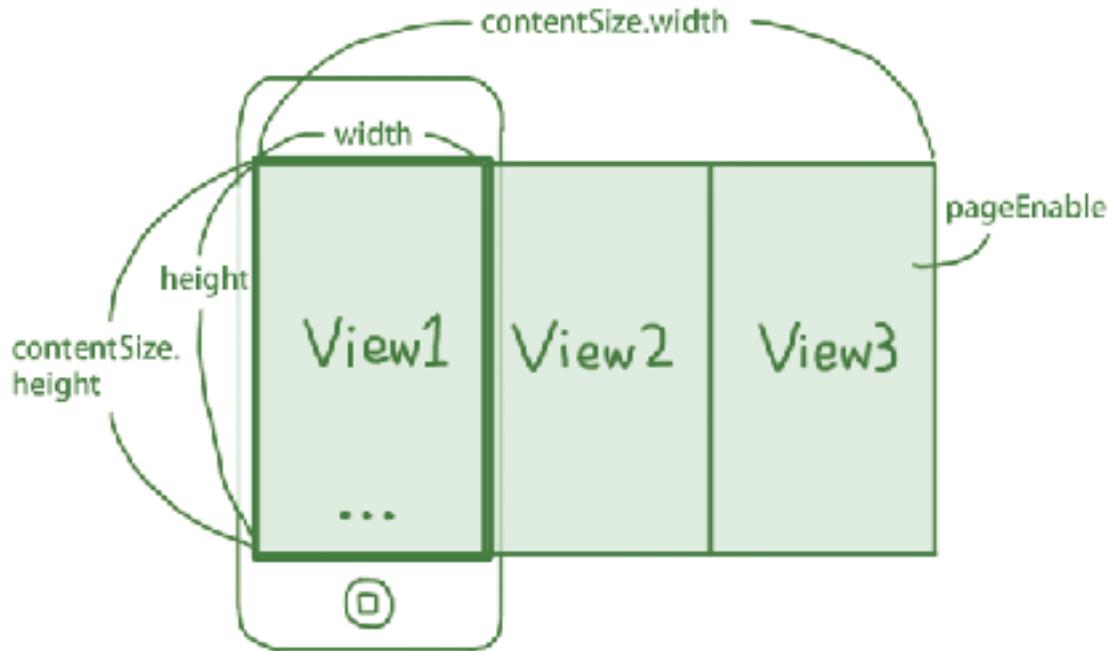
首先先讓ViewController類別服從UIScrollViewDelegate協定。在viewDidLoad方法裡設定ScrollView最大與最小縮放比例。最後覆寫UIScrollViewDelegate中的viewForZoomingInScrollView方法，回傳要放大縮小的視圖。範例中要放大縮小的是圖片myImageView。執行之後，就可以用兩根手指放大縮小畫面上的圖片。

(模擬器要模擬兩根手指的動作的話，請按著鍵盤上的option按鍵，就可以模擬兩根手指的動作。)

[用程式碼來產生UIScrollView]

範例程式碼同時記錄了如何直接用程式碼生成UIScrollView。如果要使用程式碼生成UIScrollView的話，請參考本單元的範例程式碼。

使用 UIPageControl 與 UIScrollView 製作畫面切換的效果



[問題]如何使用UIPageControl與UIScrollView製作畫面切換的效果

[解答]設定UIScrollView的isPagingEnabled與contentOffset屬性

[範例程式碼]HelloScrollViewWithPageControl

[前製設定]

1. 依照[使用ScrollView顯示比螢幕大的內容]解答的前4個步驟開啟一個新的專案，並且在畫面上加入UIScrollView。
2. 請跳到[使用ScrollView顯示比螢幕大的內容]解答的第8個步驟，把UIScrollView拉進到程式碼中。
3. 然後在ViewController類別中，加入下面的程式碼：

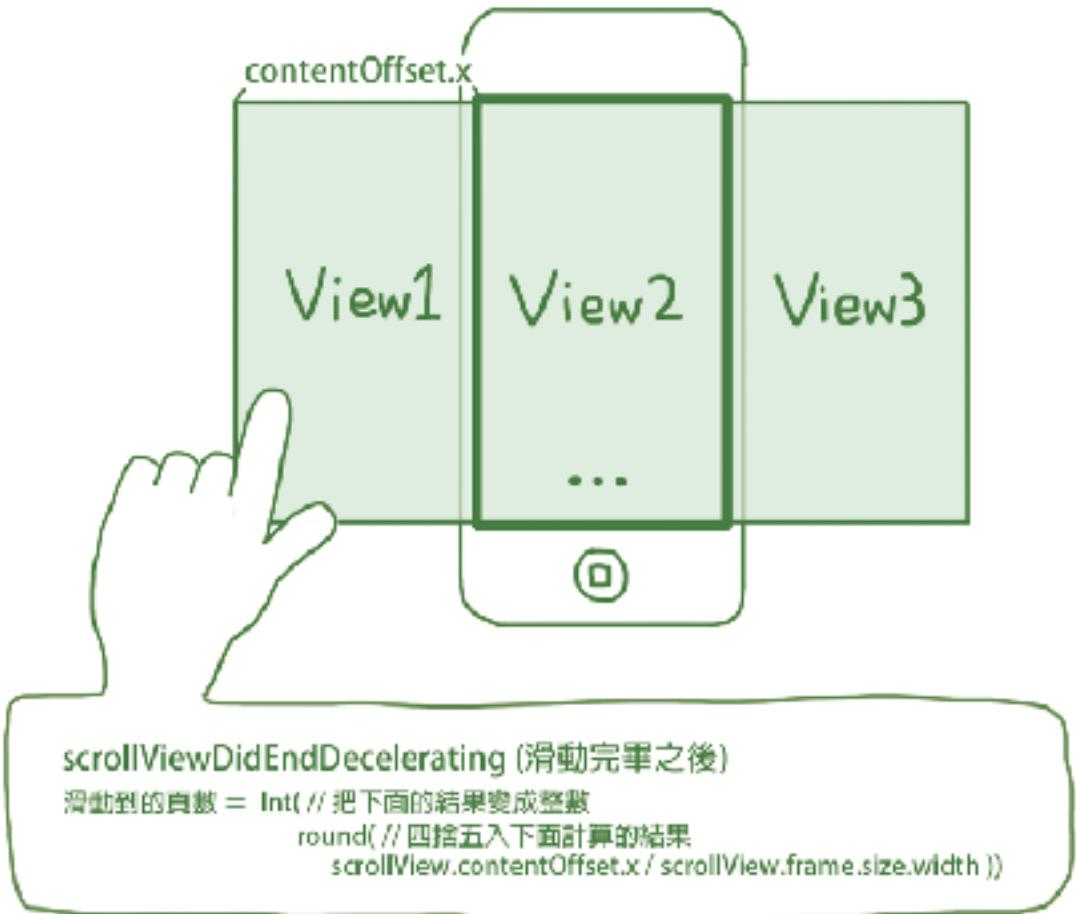
```

override func viewDidLoad() {
    //設定contentSize.width
    myScrollView.contentSize.width = myScrollView.frame.width * 3
    //設定三個畫面的顏色
    let viewColors = [UIColor.red,UIColor.green,UIColor.blue]
    //在ScrollView上加入三個畫面
    for i in 0..<3{
        let oneView = UIView(
            frame: CGRect(
                x: CGFloat(i) * myScrollView.frame.size.width,
                y: 0,
                width: myScrollView.frame.size.width,
                height: myScrollView.frame.size.height))
        oneView.backgroundColor = viewColors[i]
        myScrollView.addSubview(oneView)
    }
    //設定pageEnable
    myScrollView.isPagingEnabled = true
    //設定bounce
    myScrollView.bounces = false
}

```

- 4.本範例要做出像問題示意圖一樣的效果：在ScrollView上加入三個跟手機畫面一樣大的UIView。而Storyboard已經設定ScrollView跟畫面一樣大，所以上面的程式碼，一開始先設定ScrollView的ContentSize.width為ScrollView寬度的三倍。這樣設定，稍後才可以容納三個UIView。
- 5.為了區別三個UIView，設定三個畫面分別要填入不同的顏色。把這三個顏色存在viewColors陣列中。
- 6.利用for迴圈建立三個UIView，依序貼進ScrollView上。
- 7.程式做到這邊，編譯執行程式會發現已經可以透過滑動檢視三個不同的顏色的UIView了。不過如果設定pagingEnable為true的重新執行程式的話，會做出每次滑動就剛好滑動一頁、好像在看電子書不同頁面的效果。
- 8.設定UIScrollView的bounces屬性為false，可以避免看到白色底色與滑動到邊際彈跳的效果。

[配合UIPageControl]



1.如果要搭配`PageControl`讓`ScrollView`在滑動過後，可以在`PageControl`顯示目前的頁面。要先加入`PageControl`到畫面上。請先在`ViewController`類別中，加入`PageControl`屬性：

```
var codePageControl: UIPageControl!
```

2.繼續在`ViewController`類別的`viewDidLayoutSubviews`方法裡、已經寫好程式碼的後面，加入下面的程式碼：

```

//建立PageControl
codePageControl = UIPageControl(frame:
    CGRect(x: view.frame.size.width/2 - 30,
           y: view.frame.size.height - 50, width: 60, height: 37))
//設定PageControl顏色
codePageControl.pageIndicatorTintColor = UIColor.black
//設定目前頁面的顏色
codePageControl.currentPageIndicatorTintColor = UIColor.lightGray
//設定總頁數
codePageControl.numberOfPages = 3
//設定目前的頁數
codePageControl.currentPage = 0
//設定改動頁數後要執行的方法
codePageControl.addTarget(self, action:
    #selector(ViewController.codePageControlAction(sender:)),
                           for: .valueChanged)
//把PageControl加到畫面上
view.addSubview(codePageControl)

```

3.上面的程式碼建立了一個UIPageControl，把這個UIPageControl存入屬性codePageControl中。調整顏色、設定頁數，與改動頁數要執行的方法後，把pageControl加到畫面上。之後就可以用這個元件顯示頁數，改動pageControl的頁數會執行codePageControlAction方法。請在ViewController類別裡面加入這個方法：

```

func codePageControlAction(sender: UIPageControl) {
    let currentPageNumber = sender.currentPage //得到選取的頁面
    let width = myScrollView.frame.size.width
    //把ScrollView的寬度存在width變數之後，依照目前的頁面算出需要偏離的距離
    let offset = CGPoint(x: width * CGFloat(currentPageNumber), y: 0)
    //讓ScrollView移動到正確的位置
    myScrollView.setContentOffset(offset, animated: true)
}

```

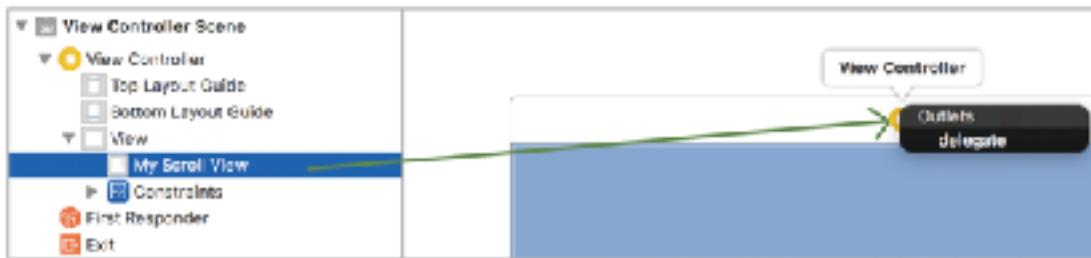
4.當使用者改動pageControl的頁數執行codePageControlAction方法時，會先取得更改後的頁數。算出該頁數的位置，然後將ScrollView移動到正確的位置顯示該頁的內容。

UIScrollView有一個叫做contentOffset的屬性。如示意圖，contentOffset是UIScrollView顯示內容(content)的頂點跟外框(frame)頂點的偏移量。算出需要偏移的距離後，如上使用setContentOffset方法就可以移動ScrollView去顯示正確的內容。

5.以上是透過更動pageControl的頁數更改ScrollView顯示頁面的方法。如果想要反過來，在滑動ScrollView後，依照滑動的結果更改pageControl顯示的頁數的話，要

- a) 設定 ViewController 類別服從 UIScrollViewDelegate 的 Protocol
- b) 讓 ViewController 類別去擔任 UIScrollView 的委派工作(delegate)。如此，ViewController 類別中，就可以
- c) 覆寫 scrollViewDidEndDecelerating 方法，在使用者滑動 UIScrollView 時，做出後續的動作、更改 pageControl 顯示的頁數。

請回到 Main.storyboard，選擇 ScrollView、一面按著鍵盤上面的 Control 鍵，一面連結到代表 ViewController 的 Icon。在跳出對話框中，選 delegate。這樣設定好之後，ViewController 類別就擔任起 UIScrollView 的委派工作(delegate)了。



6. 回到程式碼，先設定 ViewController 類別服從 UIScrollViewDelegate 的 Protocol。接著覆寫 scrollViewDidEndDecelerating 方法：

加入這個

```

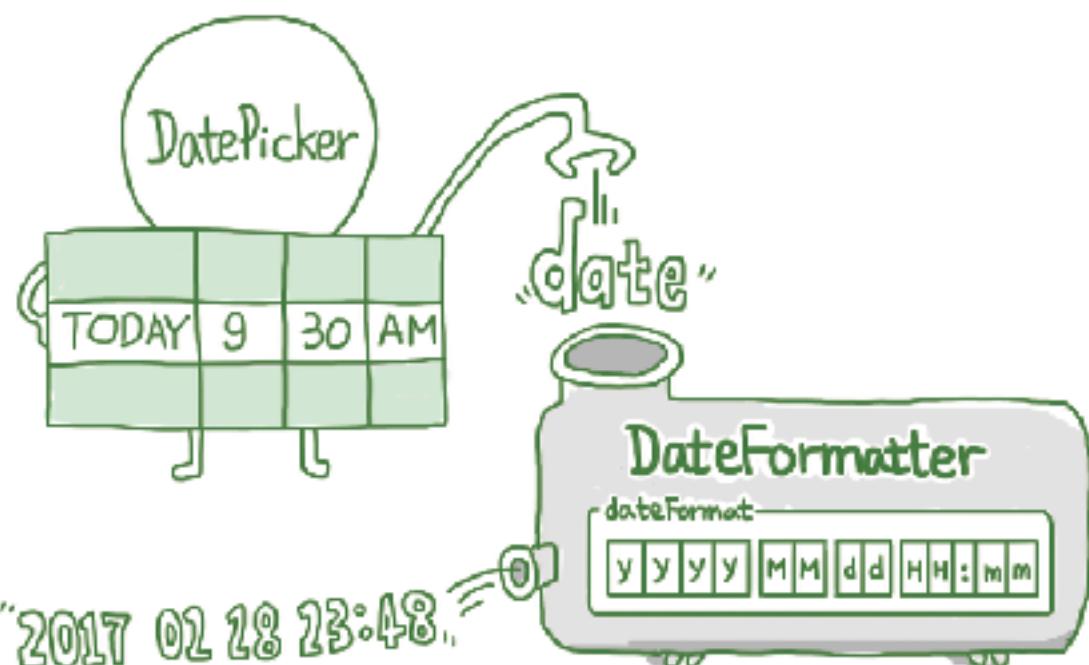
class ViewController: UIViewController, UIScrollViewDelegate {
    @IBOutlet weak var myScrollView: UIScrollView!
    var codePageControl: UIPageControl!
    func scrollViewDidEndDecelerating(_ scrollView: UIScrollView) {
        // 計算出目前滑動的頁數
        let currentPageNumber = Int(round(myScrollView.contentOffset.x / scrollView.frame.size.width))
        // 設定 pageControl 顯示的頁數
        codePageControl.currentPage = currentPageNumber
    }
}

```

設定完成之後，在使用者滑動 UIScrollView 時，就會觸發 scrollViewDidEndDecelerating 方法。在這個方法裡面，先算出目前滑動的頁數。接著把頁數設定讓 PageControl 顯示。如此，滑動換頁面的話，PageControl 就會顯示相對應的數值。

計算頁數的方法是把目前的顯示內容偏移量(contentOffset)除上 ScrollView 的寬度。把這個數字四捨五入之後，轉換成整數，就是目前翻到的頁數。

UIDatePicker選取時間



[問題]如何使用UIDatePicker選取時間

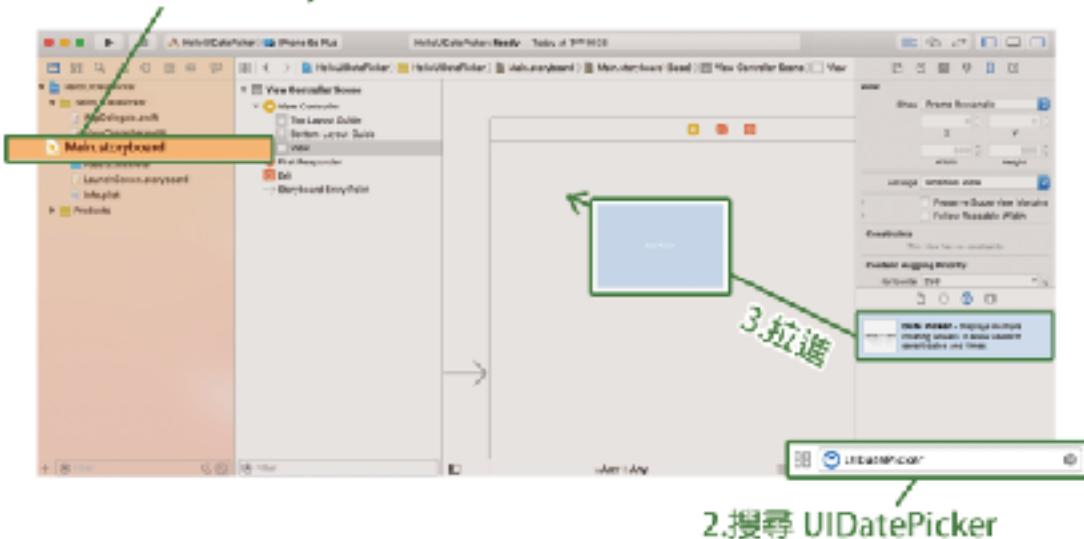
[解答]從UIDatePicker的date屬性可以得知目前的時間，再用DateFormatter類別將時間轉換成所需格式

[範例程式碼]HelloUIDatePicker

[過程說明]

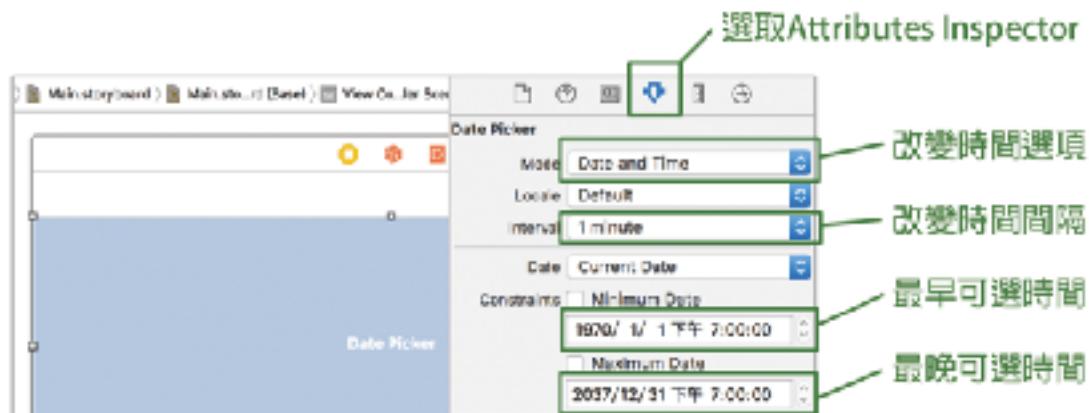
1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UIDatePicker、用滑鼠把UIDatePicker拉到畫面上。

1. 選 Main.storyboard

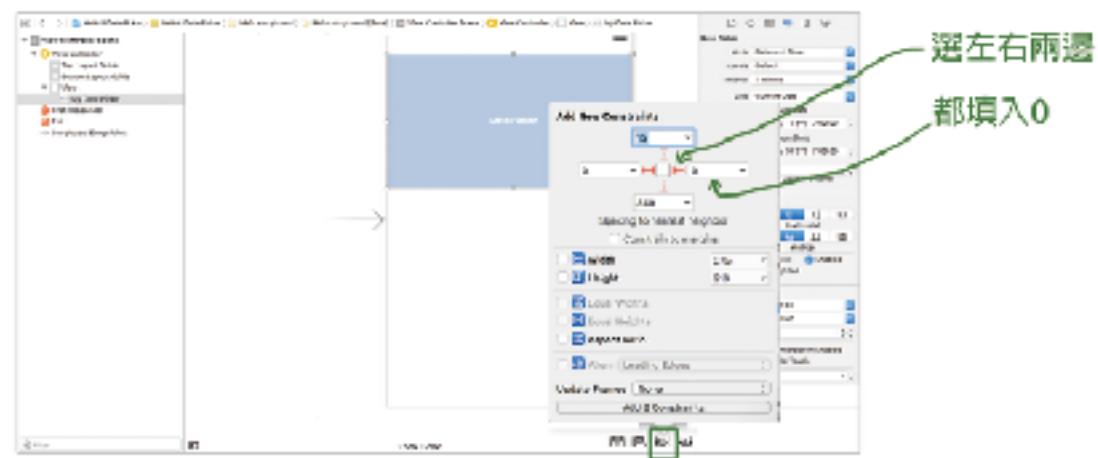


2. 搜尋 UIDatePicker

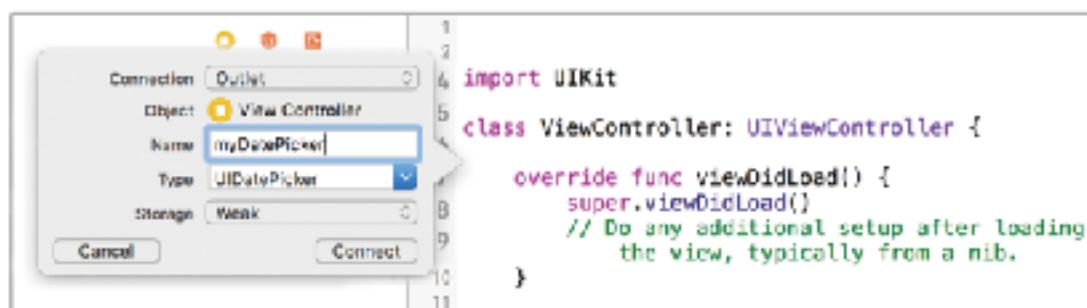
2. 點選步驟1加入的 UIDatePicker，可以在右邊欄做出各種相關設定。



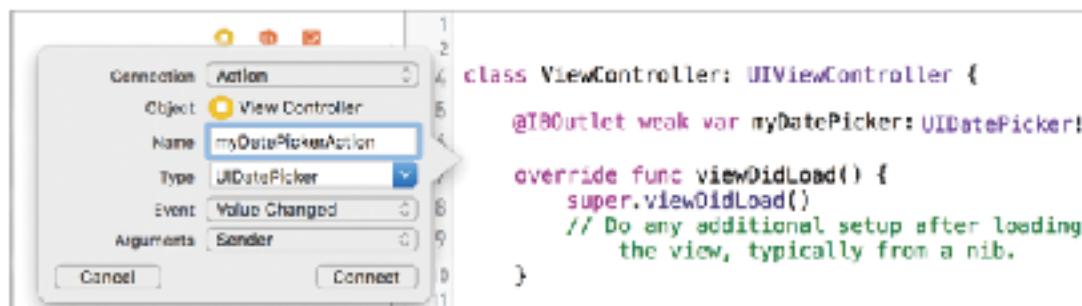
3. 選擇下方的 Pin 按鈕，可以將 DatePicker 的寬度延伸填滿畫面的左右兩邊。



4.如果要得知使用者選取的時間，要將UIDatePicker連結到程式碼中。請先選擇UIDatePicker，接著選取右上方的Assistant Editor。看到畫面分成兩半之後，一面按著鍵盤上的control鍵、一面連結UIDatePicker到程式碼中。跳出的對話框裡，幫UIDatePicker命名成myDatePicker。



5.重複步驟4的動作，再次連結UIDatePicker到程式碼中。只不過這次跳出對話框裡，在Connection選擇Action、幫這個UIDatePicker觸發的事件(IBAction)命名成myDatePickerAction，並把Type選擇成UIDatePicker。連結完成後，使用者撥動UIDatePicker、改動選取時間，就會觸發myDatePickerAction方法。



6.回到程式碼，在ViewController類別裡的myDatePickerAction方法中，寫入下面的程式碼：

```
@IBAction func myDatePickerAction(_ sender: UIDatePicker) {
    let dateFormatter = DateFormatter()
    dateFormatter.dateFormat = "yyyy MM dd EE HH:mm"
    print(dateFormatter.string(from: myDatePicker.date))
}
```

7.使用者撥動DatePicker觸發myDatePickerAction方法時，從DatePicker的date屬性就可以知道選取的時間。而透過DateFormatter類別處理後，可以為時間設定顯示格式。範例中，先建立了DateFormatter類別的物件，設定其dateFormat屬性，再以想要顯示的時間為參數，呼叫DateFormatter的stringFromDate方法，就可以以預設的格式，得知使用者選取的日期與時間。

以下整理了制訂時間格式常用的關鍵詞：

yy(西元年末兩碼)	yyyy(完整西元年份)
MMMM(完整月份)	MMM(簡寫月份)
MM(以數字表示月份)	dd(日期)
EEEE(今天星期幾)	EE(簡寫今天星期幾)
HH(24小時製制顯示小時)	hh(12小時製制顯示小時)
mm(分)	ss(秒)

範例中設定顯示日期的格式是「yyyy MM dd EE HH:mm」，所以會依不同的時間，顯示類似「2017 03 04 Fri 08:00」這樣的結果。

【用程式碼來產生UIDatePicker】

範例程式碼同時記錄了如何直接用程式碼生成UIDatePicker。如果要使用程式碼生成UIDatePicker的話，請參考本單元的範例程式碼。

UIPickerView選取資料



【問題】如何使用UIPickerView選取資料

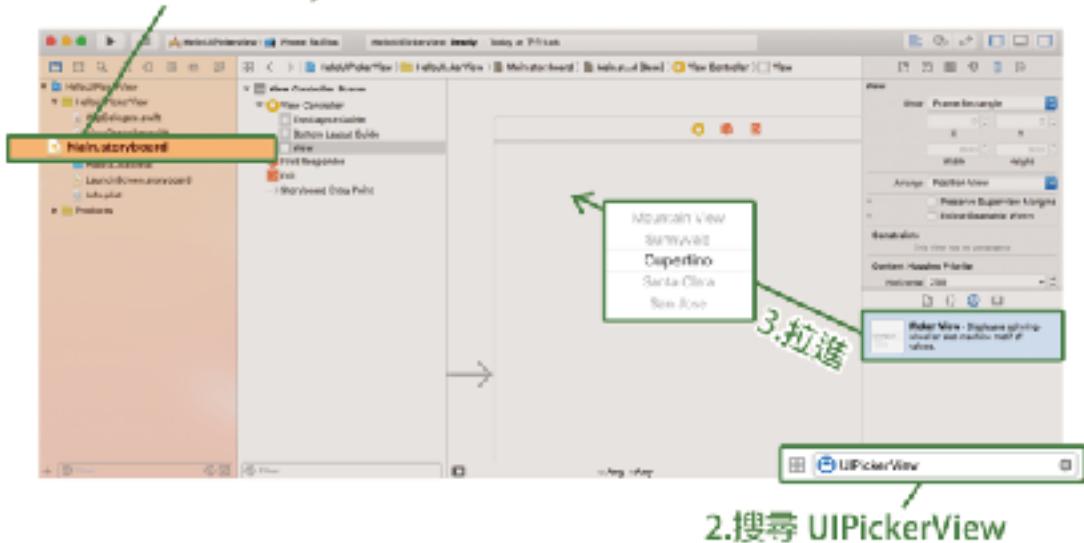
【解答】透過實做UIPickerView的兩個協定來讓UIPickerView顯示資料，並且設定選取資料後要執行的程式碼。

【範例程式碼】HelloUIPickerView

【製作兩個類別的PickerView】

1. 請先新增一個Single View Application。新增之後，在左邊欄點選 Main.storyboard。然後在右下方搜尋UIPickerView、用滑鼠把此元件拉到畫面上。

1. 選 Main.storyboard

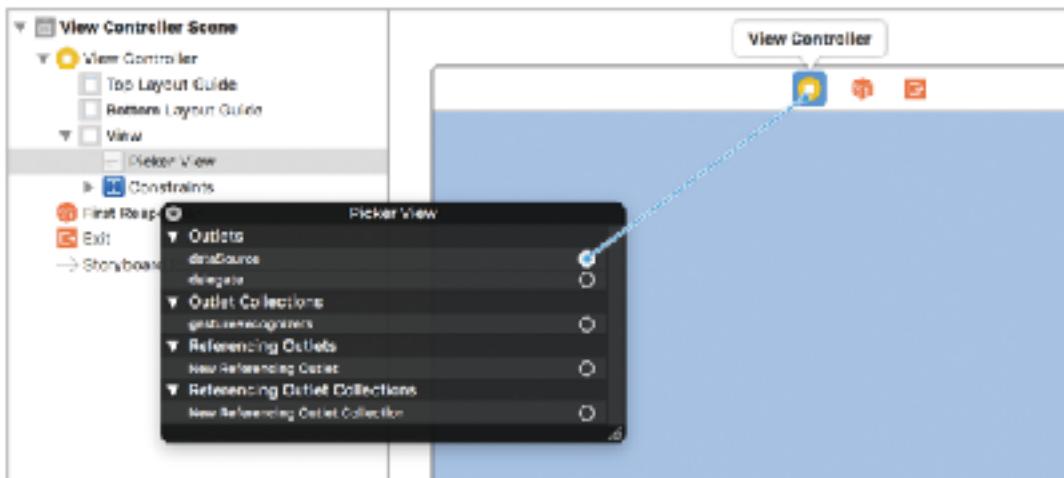


2. 搜尋 UIPickerView

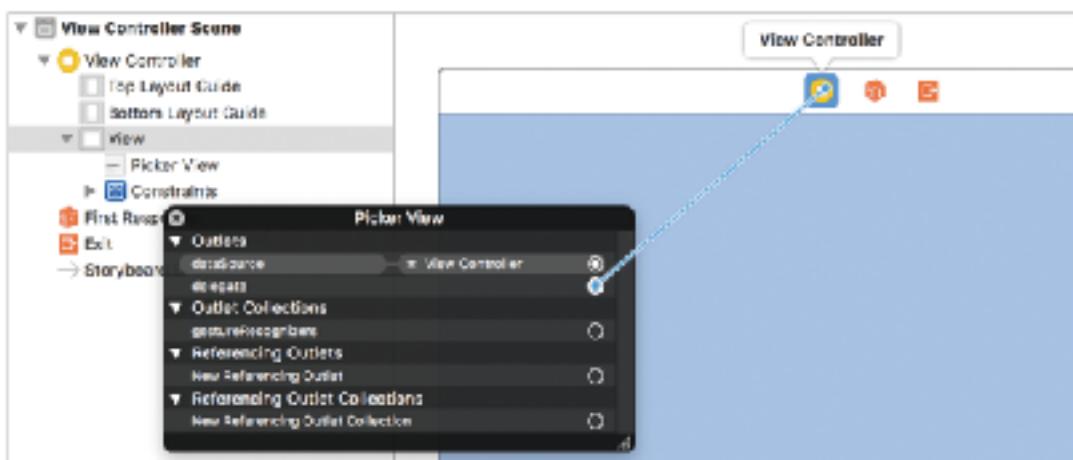
2. 先把 UIPickerView拉到畫面的下面，讓PickerView的左邊、下面，跟右邊緊貼著邊際。接著選擇下方的Pin按鈕，設定PickerView的大小。請如下圖所示，讓PickerView距離左邊、下面，跟右邊的距離都是0。這樣設定的話，執行程式時，該元件就會固定在螢幕下方，並且寬度填滿整個螢幕。



3. 此時如果執行程式，PickerView不會顯示任何的資料。UIPickerView顯示的資料需要服從UIPickerViewDataSource與UIPickerViewDelegate協定的類別來提供。如果要PickerView顯示資料的話，請先選PickerView、按右鍵。把dataSource連結到代表ViewController的Icon。



4. 請繼續把delegate也連結到代表ViewController的 Icon。連結完成代表讓 ViewController類別來提供顯示的資料，也同時讓ViewController負責設定選取資料後要執行的程式碼。



5. PickerView提供使用者選取各種資訊，完成程式後，會做出如下的 PickerView：

第一個 component
第二個 component

1	
2	
3	apple
4	banana
5	mango
6	watermelon
...	

Picker 選項的類別叫做 component。範例中，要做有兩個 component 的 Picker

?

PickerView可以建立單一或是多種類別的選項。每個類別叫做一個 component。上面的示意圖中，有兩個component。在範例中，第一個類別要顯示1~10的字串，第二個類別要顯示四種水果。

6. 在左邊欄選取ViewController.swift檔案。請把ViewController類別修改成下面的樣子：

不要忘了加入這兩個
 /)\

```
class ViewController: UIViewController, UIPickerViewDataSource, UIPickerViewDelegate {
    //首先先把要顯示的資料分別存在兩個Array
    let numberArray = ["1", "2", "3", "4", "5", "6", "7", "8", "9", "10"]
    let fruitArray = ["apple", "banana", "mango", "watermelon"]

    func numberOfComponents(in pickerView: UIPickerView) -> Int {
        return 2 //有多少個 component
    }

    func pickerView(_ pickerView: UIPickerView,
                   numberOfRowsInComponent component: Int) -> Int {
        //設定每個 component 有幾列
        if component == 0{
            return numberArray.count //第一個Component要顯示的數量
        }else{
            return fruitArray.count //第二個Component要顯示的數量
        }
    }

    func pickerView(_ pickerView: UIPickerView,
                   titleForRow row: Int, forComponent component: Int) -> String? {
        if component == 0{
            return numberArray[row] //第一個Component要顯示的文字
        }else{
            return fruitArray[row] //第二個Component要顯示的文字
        }
    }

    func pickerView(_ pickerView: UIPickerView,
                   didSelectRow row: Int, inComponent component: Int) {
        if component == 0{
            print("number: \(numberArray[row])") //點擊第一個Component
        }else{
            print("fruit: \(fruitArray[row])") //點擊第二個Component
        }
    }
}
```

7. 上面的程式碼中，首先讓ViewController類別服從UIPickerViewDataSource與UIPickerViewDelegate協定。然後設定要顯示資料。第一個component要顯示的10個數字，存在numberArray陣列中；第二個component要顯示的4個水果名

稱存在fruitArray陣列中。

8. ViewController類別實做了UIPickerViewDataSource協定的兩個方法：
numberOfComponents(in pickerView:)方法回答了PickerView有幾類的選項(有幾個component)；pickerView numberOfRowsInComponent方法回答了每個component有幾列。

9. ViewController類別同時實做了UIPickerViewDelegate協定的兩個方法：
pickerView titleForRow forComponent方法設定每個component要顯示的文字；pickerView didSelectRow inComponent方法設定使用者選取某個選項後要做的事情。目前就只印出了選項。

【製作一個類別的UIPickerView】

介紹過了兩個類別的UIPickerView作法，這種作法可以推增到三個或多個類別。如果只想要製作一個類別(component)的UIPickerView，請看本範例的程式碼。程式碼有詳細紀錄如何製作一個類別的UIPickerView。

【用程式碼來產生UIPickerView】

範例程式碼同時記錄了如何直接用程式碼生成UIPickerView。如果要使用程式碼生成UIPickerView的話，請參考本單元的範例程式碼。

UIAlertController警告控制器



【問題】如何製作跳出來的警告控制器

【解答】產生UIAlertController，就可以做出跳出警告控制器的效果

【範例程式碼】HelloUIAlertController

【過程說明】

如圖，警告控制器UIAlertController是由標題(title)跟訊息(Message)組合而成。警告控制器的按鈕是UIAlertAction類別的物件。把按鈕加入警告控制器，再把警告控制器推出，使用者就可看到跳出的警告訊息。

1. 請依照本章[使用按鈕]作法的前4個步驟，製作出一個按鈕，並且把觸發按鈕的動作連結到程式碼。在按下按鍵觸發的buttonPressed方法中寫下程式碼：

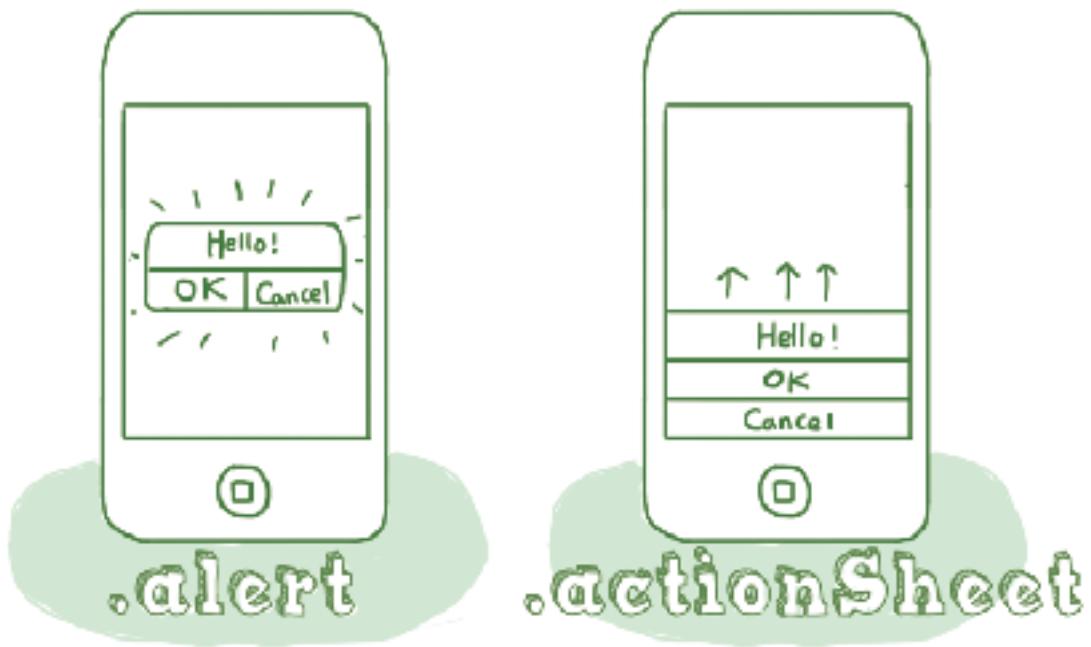
```

@IBAction func buttonPressed(_ sender: UIButton) {
    //產生AlertController
    let myAlert = UIAlertController(title: "Hello",
                                    message: "How are you", preferredStyle: .alert)
    //產生第一顆按鈕
    let okAction = UIAlertAction(title: "Fine",
                                style: .default, handler: {
        (action:UIAlertAction) -> () in
        print("fine")
        self.dismiss(animated: true, completion: nil)
    })
    //產生第二顆按鈕
    let cancelAction = UIAlertAction(title: "So So",
                                    style: .default, handler: {
        (action:UIAlertAction) -> () in
        print("so so")
        self.dismiss(animated: true, completion: nil)
    })
    //把第一顆按鈕加到警告控制器
    myAlert.addAction(okAction)
    //把第二顆按鈕加到警告控制器
    myAlert.addAction(cancelAction)
    //推出警告控制器
    self.present(myAlert, animated: true, completion: nil)
}

```

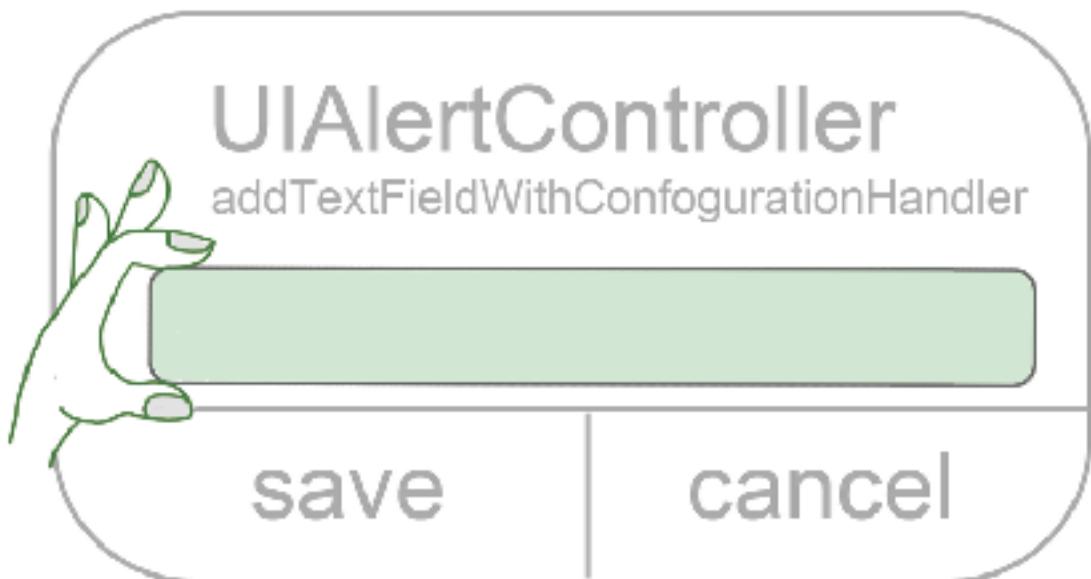
- 範例的程式碼先產生一個UIAlertController。接著連續呼叫兩次UIAlertAction的初始化方法產生兩顆按鈕，使用UIAlertController的addAction方法，分別把兩顆按鈕加到警告控制器。接著呼叫UIViewController的present方法，把警告控制器推出讓使用者看到。
- 產生UIAlertController的方法中，填入要警告的標題(title)、訊息(message)，與偏好風格(preferredStyle)。
- 產生按鈕UIAlertAction的方法中，填入按鈕的文字(title)、按鈕的風格(style)。最後一個參數是一個Closure，把按下該按鈕後要執行的程式碼寫在其中。兩個按鈕按下後除了分別顯示不同的文字，還呼叫UIViewController類別的dismiss方法，將警告控制器從畫面上移除。

[滑出式功能表(Action Sheet)]



除了從畫面中間跳出的警告控制器以外，還有一種是從手機下方跳出的選單 (Action sheet)。如果要做出這種滑出式功能表，只需修改產生警告控制器程式碼中的偏好風格。把偏好風格設定成「.alert」的話，會出現從畫面中間跳出的警告控制器；把偏好風格改成「.actionSheet」，會出現從下方跳出的選單。

【有文字輸入框的警告控制器】



警告控制器除了加上按鈕(UIAlertAction)以外，還可以加入文字輸入框。請參考下面的程式碼：

```

@IBAction func buttonPressed(sender: UIButton) {
    //產生AlertController
    let myAlert = UIAlertController(title: "Hello",
                                    message: "Enter your name", preferredStyle: .alert)
    //產生第一顆按鈕
    let saveAction = UIAlertAction(title: "Save", style: .default,
                                    handler: {
        (action:UIAlertAction) -> () in
        let alertTextField = myAlert.textFields![0] as UITextField
        print(alertTextField.text!)
        self.dismiss(animated: true, completion: nil)
    })
    //產生第二顆按鈕
    let cancelAction = UIAlertAction(title: "Cancel",
                                    style: .default, handler: {
        (action:UIAlertAction) -> () in
        self.dismiss(animated: true, completion: nil)
    })

    //加入文字輸入框
    myAlert.addTextField(configurationHandler: {
        (textField:UITextField!) -> Void in
        textField.placeholder = "Enter your name here!"
    })

    //把第一顆按鈕加到警告控制器
    myAlert.addAction(saveAction)
    //把第二顆按鈕加到警告控制器
    myAlert.addAction(cancelAction)
    //推出警告控制器
    self.present(myAlert, animated: true, completion: nil)
                                            completion: nil)
}

```

- 1.如果要在警告控制器上加上文字輸入框的話，可以使用UIAlertController的addTextField(configurationHandler:)方法加入。在其中的Closure參數中，可以設定文字輸入框預設placeholder的文字。
- 2.想要得到使用者在文字輸入框的文字，就使用UIAlertController的textFields屬性。這個屬性是一個陣列。範例中只有一個文字輸入框，textFields陣列的第一個成員，就是警告控制器的文字輸入框。把這個UITextField存在常數alertTextField中，從alertTextField的text屬性，就可以得知使用者輸入的文字。

以浮動畫面顯示資訊(Popover)



[問題]如何製作浮動畫面來顯示資訊

[解答]在Storyboard設定新的UIViewController，設定該元件為浮動畫面來顯示資訊

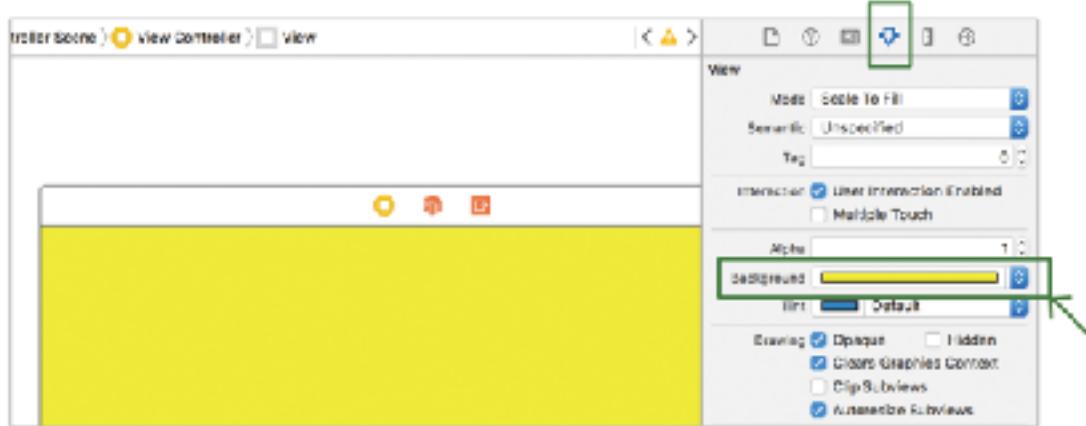
[範例程式碼]HelloPopover

[過程說明]

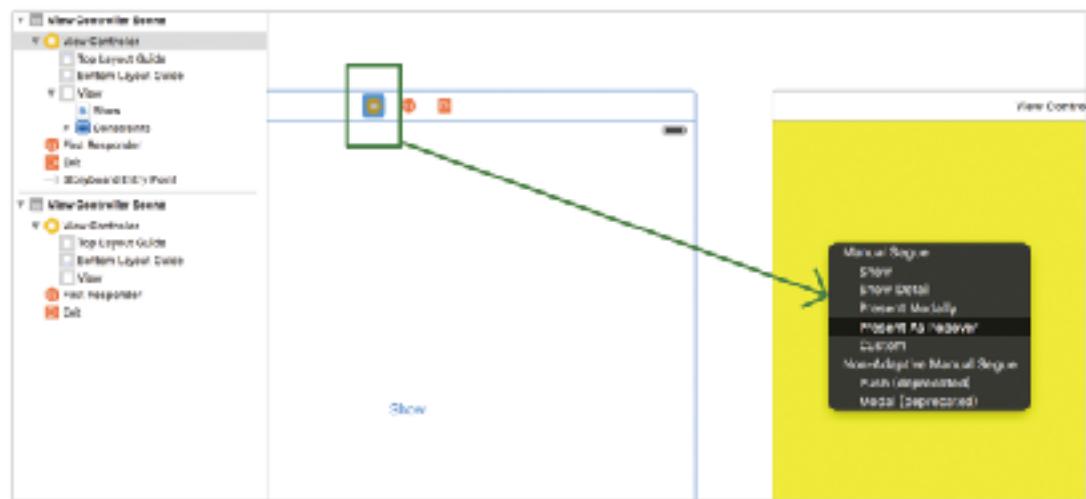
1. 請依照本章[使用按鈕]作法的前4個步驟，製作出一個按鈕，並且把觸發按鈕的動作連結到程式碼，讓按下按鈕會執行buttonPressed方法中的程式碼。本範例目標是做出按下按鈕後，從按鈕上產生浮動畫面的效果。
2. 請打開Storyboard，在右下方搜尋UIViewController，把UIViewController拉到Storyboard空著的地方。這個UIViewController就是範例中稍後要呈現的浮動畫面。



3. 為了區別此UIViewController，請把底色設成黃色。(在做您的專案時，不需跟範例一樣把底色設成黃色。請依情況設定，加入其他使用者可以互動的元素，比方說UILabel、UITextField等等。)

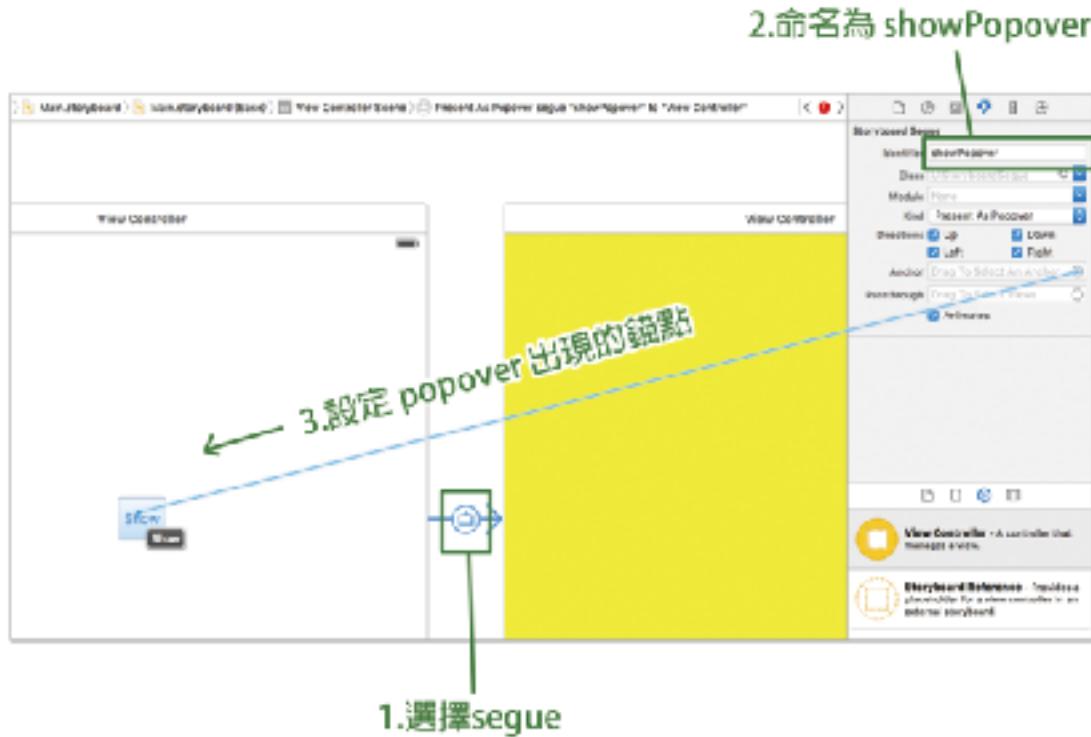


4. 選取原本代表ViewController類別的Icon，一面按著鍵盤的Control鍵，一面連結要做為Popover的UIViewController。跳出的對話框中，選Present as Popover。這樣就產生了一個segue。



5. 選取剛剛產生的segue。在右邊欄可以看到很多相關的設定：請把這個segue

命名成showPopover。並且選取Anchor一欄的連結點，把這個連結點連結到按鈕上。意思是，浮動視窗會從這個按鈕上跳出。而Directions區塊可以設定浮動畫面相對於按鈕的位置。



6.回到ViewController類別的程式碼，在按下按鈕觸發的buttonPressed方法寫下下面的一行程式碼、選用iPad來執行程式，就會在按下按鈕後看到浮動視窗(popover)出現了。(注意：要選擇iPad執行)

```
@IBAction func buttonPressed(_ sender: UIButton) {  
    performSegue(withIdentifier: "showPopover", sender: nil)  
}
```

7.不過同樣的程式碼，使用iPhone7plus、iPhone7、iPhone SE或iPhone4來執行，會發現浮動視窗會以全螢幕呈現。要讓Popover在這些機型也可以用浮動的方式呈現，請先

- a)讓ViewController類別服從UIPopoverPresentationControllerDelegate協定。
- b)加入下面的程式碼：

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "showPopover"{
        let vc = segue.destination
        //設定浮動畫面大小
        vc.preferredContentSize = CGSize(width: 200, height: 100)
        //取得 popoverPresentationController
        let controller = vc.popoverPresentationController
        if controller != nil{
            //把ViewController設定為popover的delegate
            controller?.delegate = self
        }
    }
}

func adaptivePresentationStyle(for
    controller: UIPresentationController) -> UIModalPresentationStyle {
    return .none //設定style
}
```

8.上面的程式碼中，在要執行segue前的prepareForSegue的方法中，先從即將推出的ViewController得到其popoverPresentationController屬性。將ViewController指定給這個屬性的delegate。

9.然後實做UIPopoverPresentationControllerDelegate協定的方法，設定呈現的樣式為「.none」，所有大小的機型，就都可以正常顯示浮動視窗了。

10.浮動視窗(Popover)的大小，請參考上面的範例，修改即將推出ViewController的preferredContentSize，就可以調整popover的大小。