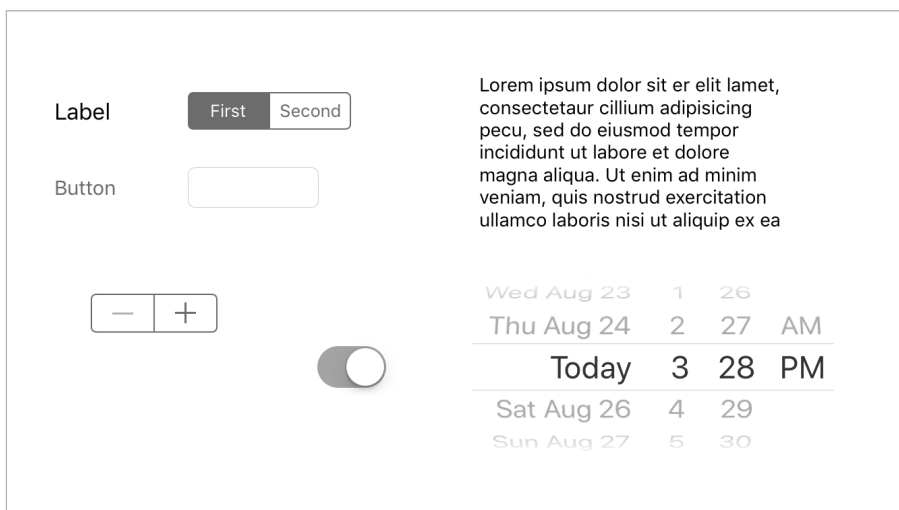


chapter

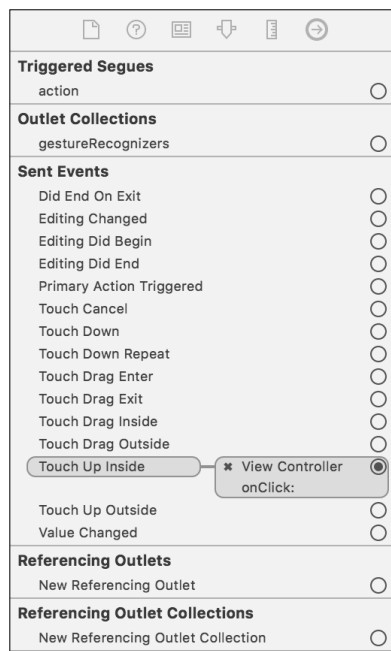
06

視覺化元件

發展 App 最重要的部分就是設計使用者介面，這也是程式設計師花最多時間的地方，從畫面的安排到元件之間的關係，在在都會影響使用者的使用意願。由於 App 是要給別人使用的，因此在開發過程中，應從使用者的角度來思考怎麼樣操作會比較方便，資料要如何呈現才能一目了然，這些都是必然要做的功課。為了讓使用者能夠有一致性的操作習慣，Xcode 提供了許多標準的視覺化的元件讓程式設計師使用，而程式設計師的工作就是在正確的時間讓使用者操作正確的元件就可以了。



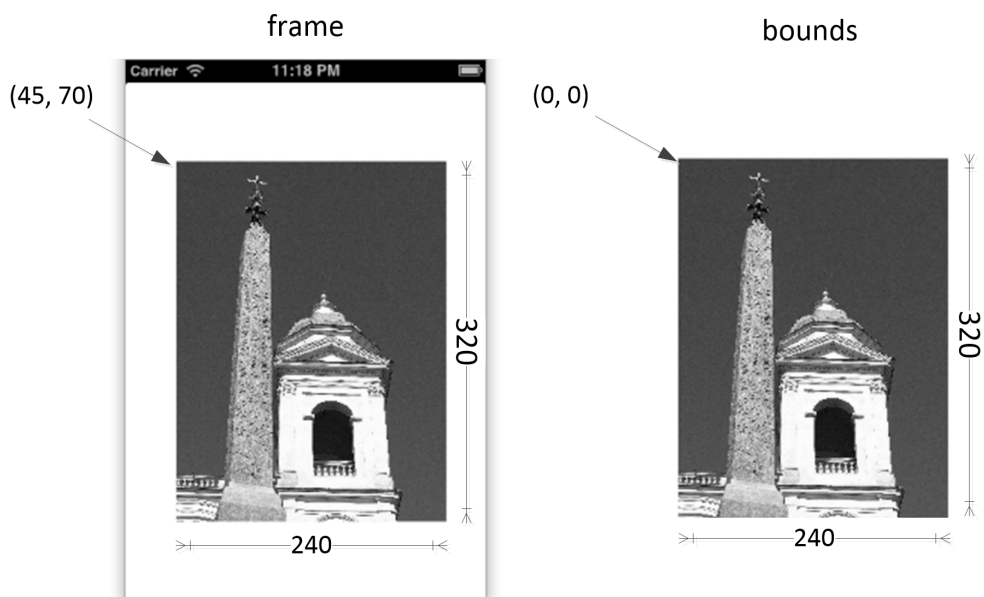
幾乎所有的視覺化元件都繼承於 `UIView` 類別，它們包含了按鈕（`UIButton`）、標籤（`UILabel`）、（圖片框 `UIImageView`）、開關（`UISwitch`）...等許多可以讓使用者操作或是呈現資料給使用者看的元件。每一個視覺化元件都有一些專屬的屬性，透過 **Attributies** 面板讓我們可以在設計階段就調整這些視覺化元件的一些屬性值，而不需要動到程式碼，例如改變顏色或是預設狀態等。除了 **Attributies** 面板外，在 **Connections** 面板中可以一覽所選的視覺化元件與 `IBAction` 方法以及 `IBOutlet` 屬性的連結狀況。要刪除視覺化元件前，最好先從這邊刪除 `IBAction` 與 `IBOutlet` 的連結，否則有時候元件刪除了，但是連結的紀錄還存在，這會導致編譯錯誤，修復起來很麻煩。



視覺化元件也具備動畫呈現效果。當 `UIView` 類別中某些屬性值改變時，這些改變可以以動畫的方式呈現。例如當改變某個按鈕的座標時，這個按鈕就會以滑動的方式從舊座標滑到新座標，而不是直接「跳」到新座標去。透過動畫，可以讓使用者在使用上有個愉快的操作經驗。下表中的屬性，就是 `UIView` 類別中具備動畫功能的屬性。

屬性	說明
<code>frame</code>	改變 view 的位置或是大小
<code>bounds</code>	改變 view 的大小
<code>center</code>	改變 view 的位置
<code>transform</code>	旋轉 view 或是放大縮小
<code>alpha</code>	改變 view 的透明度
<code>backgroundColor</code>	改變 view 的背景顏色

使用者介面的座標系統（也可稱為 **UIKit 座標系統**）是以視窗的左上角為原點，也就是 $(0, 0)$ 在左上角的位置，向右增加 x 軸的值，向下則增加 y 軸的值。每個視覺化元件有兩個跟座標有關的屬性，一個為 **frame**，另一個為 **bounds**。這兩個屬性中所指向的 **origin** 屬性儲存了這個視覺化元件的左上角座標，而 **size** 屬性則是儲存視覺化元件的長寬值。特別一提的是，**frame** 與 **bounds** 所儲存的座標系統有些不一樣。在 **frame** 中，**origin** 儲存的座標值是相對於 **superview**（也就是這個視覺化元件放在哪一個元件「裡面」，那個元件就是這個視覺化元件的 **superview**）的座標系統，而 **bounds** 則是自己的座標系統。



6-2

難易度



按鈕

➤ 先備知識：4-3 攔截事件

按鈕，`Button`（類別為 `UIButton`），用途是用來得知使用者是否用手指在螢幕上點一下。按鈕上除了使用文字提示使用者這個按鈕的用途外，還可以將文字換成圖片。按鈕本身有四種狀態，例如預設狀態、手指點下但未放開、被選擇或是禁用。一般來說，我們並不需要根據按鈕的狀態來做特別的設定，但如果我們需要設定按鈕按下去時將背景顏色換成別的顏色，這時我們就可以指定按鈕在手指點下但未放開時（`highlighted`）的背景顏色為綠色。設定按鈕上的文字或圖片，也必須告訴按鈕，文字或圖片是要在哪種狀態下來顯示。

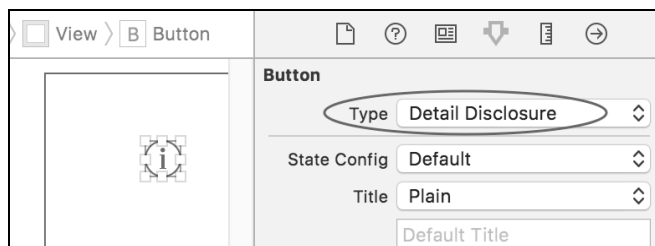


步驟與說明

1 建立 Single View App 專案。

2 開啟 `Main.storyboard`，拖放一個 `Button` 元件到 `View Controller` 上。根據先備知識，用藍線拉出按鈕的 `IBAction` 函數就可以知道使用者在此按鈕上點了一下，這裡請讀者參考先備知識，不再贅述。

3 我們可以從 `Button` 的屬性面板中更改顯示的樣式，例如將「`Type`」改成 `Detail Disclosure`，`Button` 就會顯示一個圈圈裡面有個 `i`，代表 `information` 的意思。



01

02

03

04

05

06

07

08

09

10

11

12

13

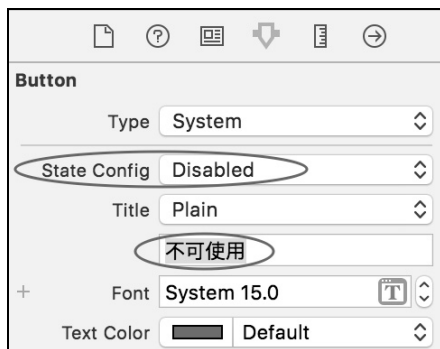
14

視覺化元件

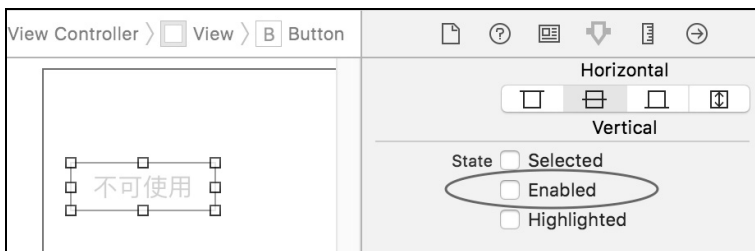
補充說明

因為 Button 的 `buttonType` 屬性為唯讀屬性，因此樣式調整只能在 Xcode 中操作。

4 除顯示型態外，Button 還有四種狀態：`default`、`highlighted`、`selected` 與 `disabled`。分別代表預設狀態、手指點下但未鬆開也未移出按鈕範圍、被選擇以及禁用。我們可以分別設定這四種狀態要顯示的文字。例如我們將狀態改為 `disabled`，然後顯示文字欄位輸入「不可使用」。



5 然後將 `Enabled` 勾勾拿掉，這時按鈕的狀態就是禁用狀態，按鈕上的文字變成設定的「不可使用」。



6 如果要在執行中變更狀態或是更改 Button 文字，程式碼如下，但要先用藍線拉出 Button 的 IBOutlet 屬性，例如 `button`。

```
class ViewController: UIViewController {  
  
    @IBOutlet weak var button: UIButton!  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
        // Do any additional setup after loading the view, typically from a nib.  
    }  
}
```

```
button.isEnabled = true
button.isSelected = true
button.setTitle("被選到了", for: .selected)
}
```

7 執行看看。



補充說明

1. 使用 `setImage(image:for:)` 來設定按鈕在何種狀態下顯示何種圖片。
2. 使用 `setBackgroundImage(image:for:)` 來設定背景圖片。
3. 使用 `backgroundColor` 屬性來設定背景顏色。

01

02

03

04

05

06

07

08

09

10

11

12

13

14

視覺化元件

6-3

難易度



分段控制

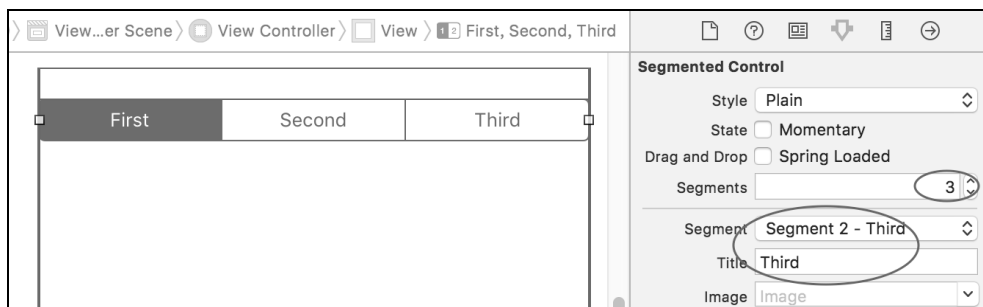
先備知識：無

分段控制，Segmented Control（類別為 `UISegmentedControl`），用來顯示像「頁籤」一樣的畫面。當使用者點選不同區段按鈕時，透過 `IBAction` 函數就可以知道使用者點選了哪一個區段。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 `Main.storyboard`，拖放一個 Segmented Control 元件到 View Controller 上。
- 3 在屬性面板上將 Segmented Control 的分段數量調整為 3 段，並且設定第 3 段的文字為 Third。



- 4 將 Xcode 畫面切換成輔助編輯模式，用藍線拉出 Segmented Control 在 `ViewController.swift` 中的 `IBAction` 函數。

```
@IBAction func click(_ sender: UISegmentedControl) {  
    // 取得點選到的分段的索引，最左側為 0  
    let index = sender.selectedSegmentIndex  
    // 根據索引取得分段上的標題文字  
    print(sender.titleForSegment(at: index)!)  
}
```

- 5 執行看看。

6-4

難易度



文字框

➤ 先備知識：無

文字框，Text Field（類別為 UITextField），目的是讓使用者可以輸入資料，因為是「文字」框，因此不論使用者輸入什麼內容，一律以 String 型態儲存在文字框的 text 屬性中。

文字框有所謂的「第一回應者（first responder）」狀態，當任何一個文字框取得第一回應者時，裝置上就會出現虛擬鍵盤，這時使用者在鍵盤上打的字就會送到擁有第一回應者的文字框中。我們可以透過在文字框上點一下，或是呼叫文字框的 becomeFirstResponder() 函數讓該文字框變成第一回應者。如果要關閉虛擬鍵盤，可以呼叫 resignFirstResponder()。

在文字框的屬性面板中設定「Clear Button」，就可以讓文字框右方出現刪除按鈕（預設是永不出現），如下圖。



此外，在屬性面板中設定鍵盤類型，可以讓只需要輸入數字的場合（例如輸入信用卡卡號），鍵盤只會出現數字讓使用者點選。這是一個貼心設計，但某些鍵盤類型在 iPhone 與 iPad 效果不同，請讀者務必在兩邊都確認過。當屬性面板中勾選了「Secure Text Entry」，在使用者輸入資料時顯示的文字就會以圈圈代替，專用於密碼輸入。

關閉鍵盤在 iPhone 上是重要的。iPad 的虛擬鍵盤本身就內建一個「關閉鍵盤」的按鍵（在最右下角），所以使用者可以根據當時需要來決定是否關閉鍵盤。但是在 iPhone 上卻沒有這個鍵，因此使用者必須靠鍵盤上的 return 鍵（不一定顯示 return，在屬性面板上設定「Return Key」類型，例如改成 Google，就會顯示「搜尋」或「Search」）來關閉鍵盤。這裡會因為 Text Field 元件為單行文字框，因此 return 鍵按下時會自動呼叫 resignFirstResponder() 然後關閉鍵盤，如果是 Text View 元件就會變成換行，若要關閉鍵盤就得另想辦法（請參考本章「鍵盤+工具列」一節）。除此之外，如果鍵盤類型是 Number Pad，這種類型的鍵盤上沒有 return 鍵，所以如果不想辦法把鍵盤關掉，鍵盤也就會一直顯示在螢幕上。

01

02

03

04

05

06

07

08

09

10

11

12

13

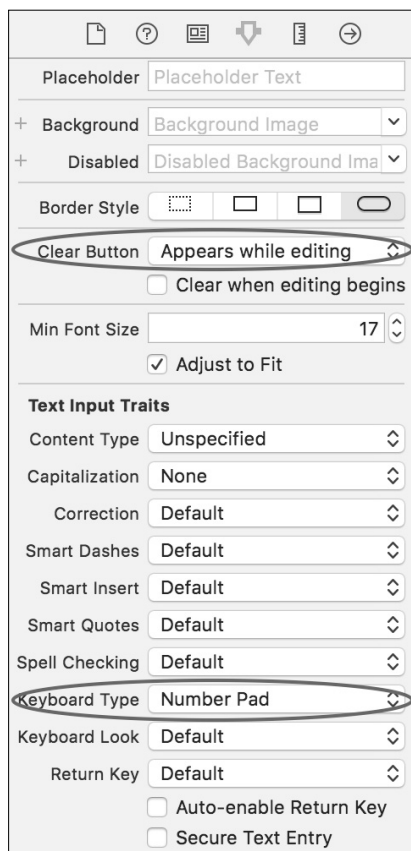
14

視覺化元件

這個單元，我們來看當鍵盤類型沒有 enter 鍵時如何關閉鍵盤。方法其實很多，這邊我們使用一個比較直覺的，就是當使用者在「別的地方」點一下時關掉鍵盤，另外一種是在鍵盤上加工具列，然後在工具列上加一個關閉按鈕，此部分請參考本章「鍵盤+工具列」一節。

步驟與說明

- 1** 建立 Single View App 專案。
- 2** 開啟 Main.storyboard，拖放一個 Text Field 元件到 View Controller 上。
- 3** 在屬性面板上將「Clear Button」改為 Appears while editing，這樣可以在文字框右邊出現刪除內容的按鈕，然後再將「Keyboard Type」改為 Number Pad。



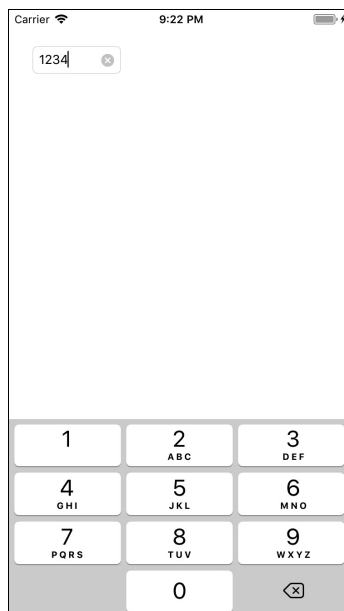
4 開啟 `ViewController.swift`，覆寫 `touchesBegan(_:with:)` 函數，在這個函數中結束 `view` 的編輯狀態，此時會讓 `view` 上面所有的 `Text Field` 結束編輯，自然也就解除了第一回應者。除此之外，也可以呼叫每一個 `Text Field` 的 `resignFirstResponder()` 函數。

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

5 上一步的程式碼會讓鍵盤立即關掉，如果我們想要讓鍵盤有一個向下滑動關掉的動畫特效，可以透過 `UIView` 來處理。下面程式碼讓鍵盤加上一個 0.3 秒的向下滑動關閉的動畫特效。

```
override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {  
    UIView.animate(withDuration: 0.3) {  
        self.view.endEditing(true)  
    }  
}
```

6 執行看看。



6-5

難易度



滑桿

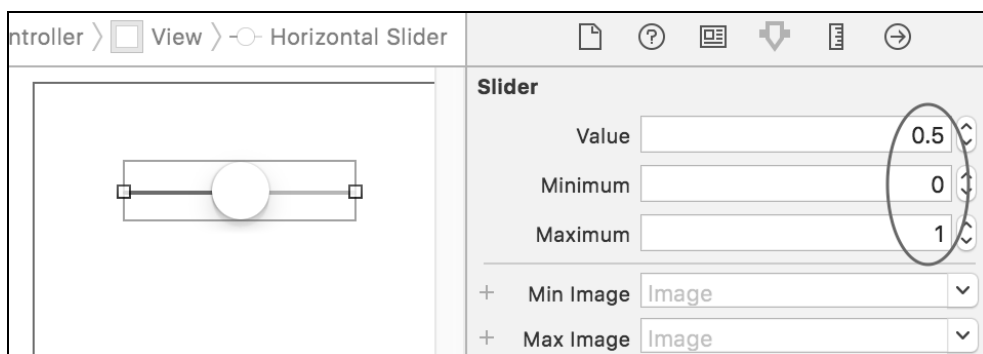
先備知識：無

滑桿，Slider（類別為 UISlider），用來讓使用者透過滑動的方式設定數值。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 Main.storyboard，拖放 Slider 元件到 View Controller 上。在屬性面板上可以設定滑桿的現行、最小與最大值。



- 3 將 Xcode 畫面切換成輔助編輯模式，在 Slider 上用藍線拉到 ViewController.swift 中，為 Slider 設定 IBAction 函數。參數型態請由 Any 改為 UISlider。

```
@IBAction func valueChanged(_ sender: UISlider) {  
    print(sender.value)  
}
```

- 4 執行看看。

0.504237

0.512712

6-6

難易度



開關

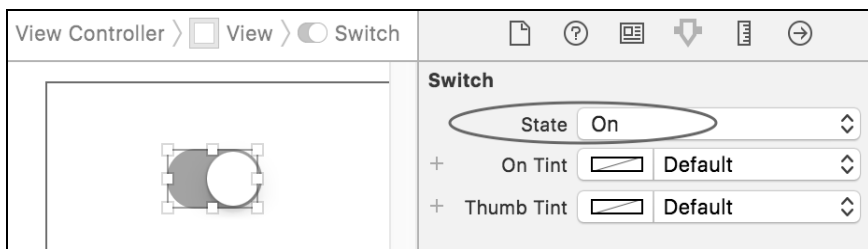
先備知識：無

開關，Switch（類別為 UISwitch），提供了一個類似開關的功能，只有 On / Off 兩種狀態。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 Main.storyboard，拖放一個 Switch 元件到 View Controller 上。在屬性面板上可以修改 Switch 元件的預設狀態，也可以改變預設顏色。



- 3 將 Xcode 畫面切換成輔助編輯模式，在 Switch 上用藍線拉到 ViewController.swift 中，為 Switch 設定 IBAction 函數。參數型態請由 Any 改為 UISwitch。

```
@IBAction func valueChanged(_ sender: UISwitch) {  
    if sender.isOn {  
        print("開")  
    } else {  
        print("關")  
    }  
}
```

- 4 執行看看。

01

02

03

04

05

06

07

08

09

10

11

12

13

14

視覺化元件

6-10

難易度



步進

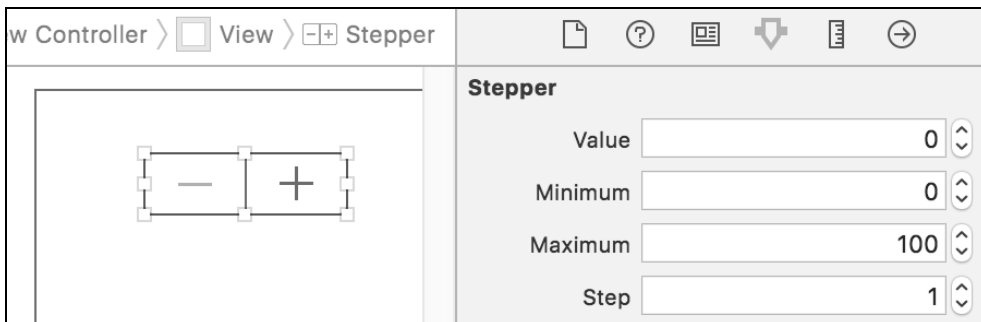
先備知識：無

步進，Stepper（類別為 `UIStepper`），透過「+」「-」按鈕來讓使用者增加或減少數值，透過這個元件，使用者就不需要用打字的方式輸入數字資料。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 Main.storyboard，將 Stepper 元件拖放到 View Controller 上。在屬性面版上可以設定 Stepper 的數值範圍以及每次的步進距離。資料型態為 Double。



- 3 將 Xcode 畫面切換成輔助編輯模式，用藍線在 `ViewController.swift` 中拉出 Stepper 元件的 `IBAction` 函數。此外，傳進去的參數型態由 `Any` 改為 `UIStepper`。在函數中我們將 Stepper 目前的步進值列印出來。

```
@IBAction func stepper(_ sender: UIStepper) {  
    print(sender.value)  
}
```

- 4 執行看看。若步進值已經到最大或最小值，「+」或「-」按鈕會變成灰色的 `disable` 狀態。

6-11

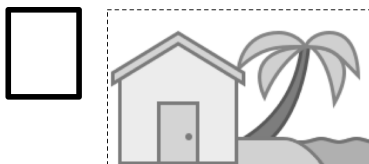
難易度



圖片框

➤ 先備知識：無

圖片框，Image View（類別為 UIImageView），用途很單純，就是用來顯示圖片。但圖片的大小通常跟圖片框大小不一樣，為了讓照片可以適當的顯示在螢幕上，我們可以調整 Image View 的顯示模式讓圖片以最適合的方式來呈現。透過 `contentMode` 屬性就可以設定圖片框要如何顯示圖片。為方便說明，我們假設下圖中左側粗線框代表圖片框，也就是 UIImageView，右邊虛線框則代表要顯示的圖片。



模式說明	參數名稱	圖例
Scale To Fill	<code>.scaleToFill</code>	
Aspect Fit	<code>.scaleAspectFit</code>	
Aspect Fill	<code>.scaleAspectFill</code>	
Redraw	<code>.redraw</code>	
Center	<code>.center</code>	

01

02

03

04

05

06

07

08

09

10

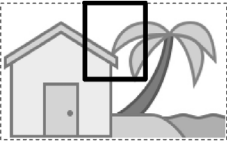

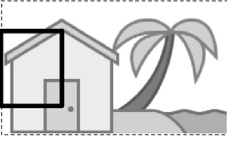
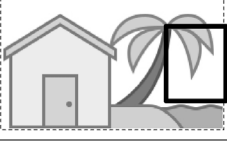
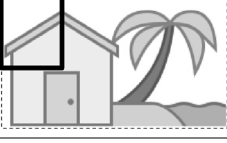
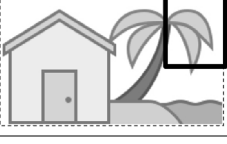
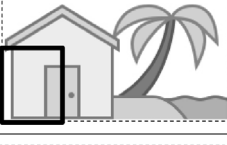
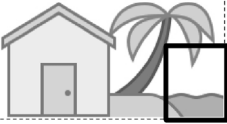
11

12

13

14

視覺化元件

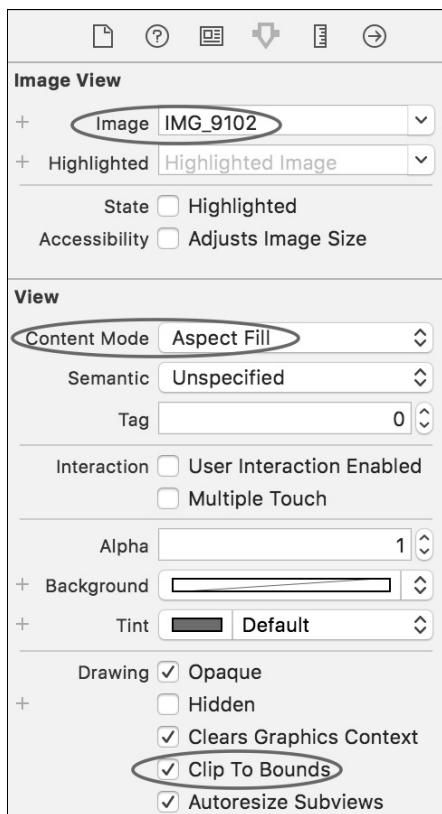
模式說明	參數名稱	圖例
Top	.top	
Bottom	.bottom	
Left	.left	
Right	.right	
Top Left	.topLeft	
Top Right	.topRight	
Bottom Left	.bottomLeft	
Bottom Right	.bottomRight	



步驟與說明

1 建立 Single View App 專案，並將一張 jpeg 圖檔加到專案中。

2 開啟 Main.storyboard，拖放一個 Image View 元件到 View Controller 上。在屬性面板中透過 Image 欄位直接載入圖片，也可以預先設定 Content Mode（同等於 contentMode 屬性）形式。最後，「Clip To Bounds」建議勾選起來，這個選項是當圖片大小超過 Image View 範圍時，將超過的部分裁減掉，如果沒有勾選，超過的部分依然會顯示在螢幕上。



3 將 Xcode 畫面切換成輔助編輯模式，在 Image View 上拉藍線拉到 ViewController.swift 上，為 Image View 設定一個 IBOutlet 屬性。

```
class ViewController: UIViewController {  
  
    @IBOutlet weak var imageView: UIImageView!
```

01

02

03

04

05

06

07

08

09

10

11

12

13

14

視覺化元件

4 在 `viewDidLoad()` 中載入圖片並顯示在 Image View 元件上。

```
override func viewDidLoad() {  
    super.viewDidLoad()  
    // Do any additional setup after loading the view, typically from a nib.  
    // 以下這三行程式碼在屬性面板中都對應欄位可以設定  
    imageView.contentMode = .scaleAspectFill  
    imageView.clipsToBounds = true  
    imageView.image = UIImage(named: "IMG_9102")  
}
```

5 執行看看。



6-12

難易度



連續播放圖片

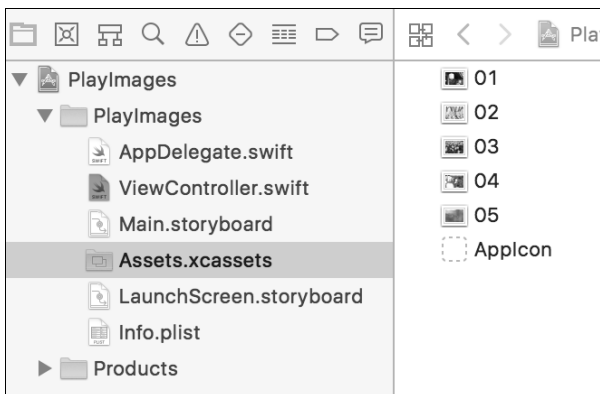
先備知識：無

Image View 元件除了顯示單一圖片外，也可以在設定的時間內連續顯示多張圖片。作法只要先將所有要顯示的圖片放到型態為 UIImage 的陣列中，然後要 Image View 在設定時間內把陣列裡面的圖片播放完即可。我們可以用這種方式來產生動畫。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 Assets.xcassets，將要顯示的圖片拖放到這個檔案中，為了之後程式處理起來方便，先把名稱改為 01、02...05。



- 3 開啟 Main.storyboard，拖放一個 Image View 元件到 View Controller 上。
- 4 將 Xcode 畫面切換成輔助編輯模式，在 Image View 元件上拉藍線到 ViewController.swift 中，為 Image View 設定一個 IBOutlet 屬性。

```
class ViewController: UIViewController {  
  
    @IBOutlet weak var imageView: UIImageView!
```

01

02

03

04

05

06

07

08

09

10

11

12

13

14

視覺化
元件

5 在 `viewDidLoad()` 中先用迴圈將 5 張圖片加到陣列中，再將這個陣列交給 `UIImage` 的 `animationImages` 屬性，然後使用 `animationDuration` 屬性設定播放時間為 10 秒，預設值為陣列中的圖片數量乘上 1/30 秒。最後呼叫 `startAnimating()` 就開始依序播放圖片。如需中途停止播放，呼叫 `stopAnimating()` 即可。

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

    var images = [UIImage]()

    // 載入所有要播放的圖片
    for i in 1...5 {
        let name = String(format: "%02d", i)
        images.append(UIImage(named: name)!)
    }

    imageView.animationImages = images
    // 設定 10 秒內播完
    imageView.animationDuration = 10
    // 設定重複 2 次
    imageView.animationRepeatCount = 2

    // 開始播放
    imageView.startAnimating()
}
```

6 執行看看。

6-14

難易度



捲軸

先備知識：無

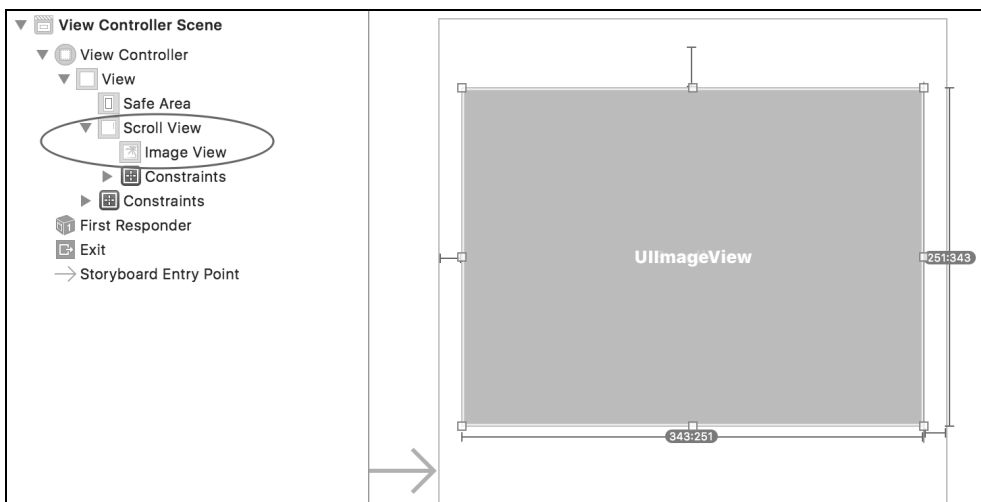
捲軸，Scroll View（類別為 UIScrollView），目的是透過兩側的捲軸或是縮放手勢來顯示範圍超過 Scroll View 的內容。例如在 Scroll View 上放置 Image View，就可以讓大圖片有捲軸捲動了。

可以捲動範圍由 Scroll View 的 `contentSize` 屬性設定，如果 `contentSize` 大小超過 Scroll View 的大小，代表要使用捲軸來呈現超過範圍的內容。由於 Scroll View 內建縮放手勢，因此可以設定 Scroll View 最大與最小倍率，預設值都是 1。

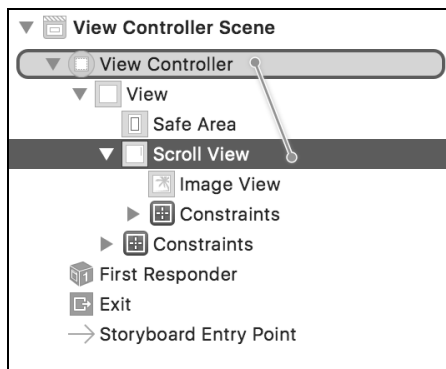


步驟與說明

- 1 建立 Single View App 專案，並在專案中加入一張 jpeg 圖檔。
- 2 開啟 Main.storyboard，拖放一個 Scroll View 元件到 View Controller 上，然後再拖放一個 Image View 元件到 Scroll View 上。注意 Image View 的位置必須是 Scroll View 的 sub view。分別設定好 Scroll View 以及 Image View 的 constraint（這邊 constraint 不好設定，需多試幾次）。



3 用藍線拉出 Scroll View 在 View Controller 的 delegate，藍線從 Scroll View 拉到 View Controller 上的黃圈圈後選擇 delegate。



4 將 Xcode 畫面切換成輔助編輯模式，分別用藍線拉出 Scroll View 與 Image View 在 ViewController.swift 中的 IBOutlet 屬性。最後讓 ViewController 符合 UIScrollViewDelegate 協定。

```
class ViewController: UIViewController, UIScrollViewDelegate {  
  
    @IBOutlet weak var scrollView: UIScrollView!  
    @IBOutlet weak var imageView: UIImageView!
```

5 在 viewDidLoad() 中載入圖片並且透過放大倍率來設定 Scroll View 的捲軸捲動範圍。

```
override func viewDidLoad() {  
    super.viewDidLoad()  
    // Do any additional setup after loading the view, typically from a nib.  
    // 載入圖片並取得圖片原始大小  
    imageView.image = UIImage(named: "IMG_8800")  
    let size = imageView.image!.size  
    // 讓 scroll view 的 content 大小與圖片實際大小一樣  
    scrollView.contentSize = size  
  
    // 若一開始想將圖片完整顯示在 scroll view 範圍內時，  
    // 必須先計算出 scroll view 中的 content 放大倍率  
    let wScale = scrollView.frame.size.width / size.width  
    let hScale = scrollView.frame.size.height / size.height  
    let scale = min(wScale, hScale) // 完整顯示圖片的放大倍率  
  
    // 設定最小、最大與目前放大倍率  
    scrollView.minimumZoomScale = scale  
    scrollView.maximumZoomScale = 0.8
```

01
02
03
04
05
06
07
08
09
10
11
12
13
14

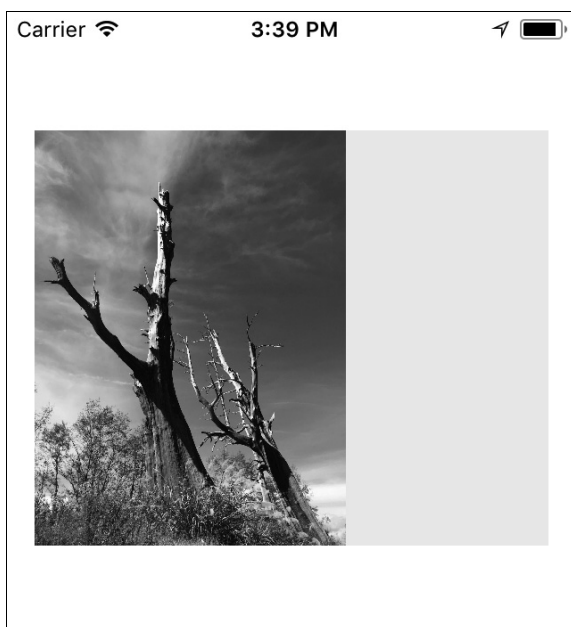
視覺化元件

```
scrollView.zoomScale = scale
}
```

6 實作 `viewForZooming(in:)`，這個函數會在使用者使用放大縮小手勢時呼叫，只要在這個函數中回傳哪個 `UIView` 元件需要放大縮小即可。

```
func viewForZooming(in scrollView: UIScrollView) -> UIView? {
    return imageView
}
```

7 先來執行看看。此時圖片會靠 `Scroll View` 的左上角，如圖所示，灰色部分為 `Scroll View`。



8 若想要將圖片移到中間，先實作一個自訂函數，在這個函數中計算 `Image View` 與 `Scroll View` 間的留空距離，然後調整留空即可。

```
func moveToCenter() {
    let imageViewSize = imageView.frame.size
    let scrollViewSize = scrollView.frame.size
    var wPadding: CGFloat = 0
    var hPadding: CGFloat = 0

    if imageViewSize.width < scrollViewSize.width {
        wPadding = (scrollViewSize.width - imageViewSize.width) / 2
    }
}
```

```
if imageViewSize.height < scrollViewSize.height {
    hPadding = (scrollViewSize.height - imageViewSize.height) / 2
}
```

```
scrollView.contentInset = UIEdgeInsets(top: hPadding, left: wPadding,
    bottom: hPadding, right: wPadding)
```

```
}
```

9 在三個函數中呼叫上一步的自訂函數。

```
override func viewDidAppear(_ animated: Bool) {
    super.viewDidAppear(animated)
    moveToCenter()
}
```

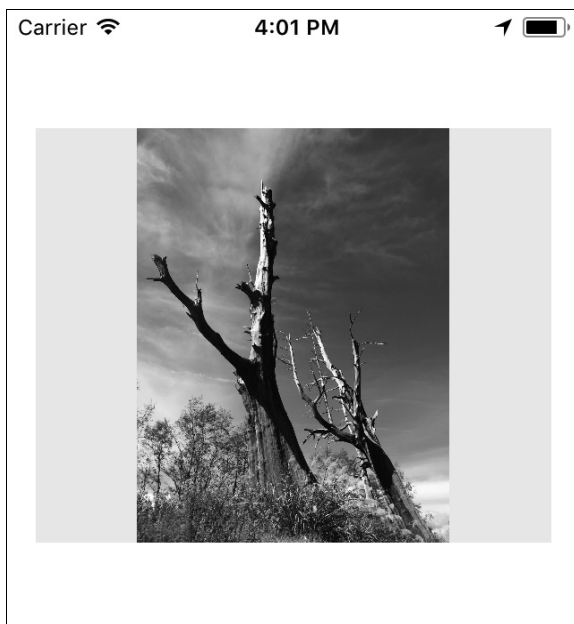
```
// 完成放大縮小時呼叫
```

```
func scrollViewDidZoom(_ scrollView: UIScrollView) {
    moveToCenter()
}
```

```
// 進行放大縮小時呼叫
```

```
func viewForZooming(in scrollView: UIScrollView) -> UIView? {
    moveToCenter()
    return imageView
}
```

10 執行看看。當圖片大小比 Scroll View 小的時候，圖片會位於中間。



6-15

難易度



選取器

先備知識：無

選取器，Picker View（類別為 `UIPickerView`），讓使用者可以從一份清單中點選需要的項目。Picker View 特殊的地方在於當清單中的項目過多時，Picker View 會將清單中的所有選項放在滾輪上，因此使用者是用滾動滾輪來瀏覽各個項目而不是透過捲軸。

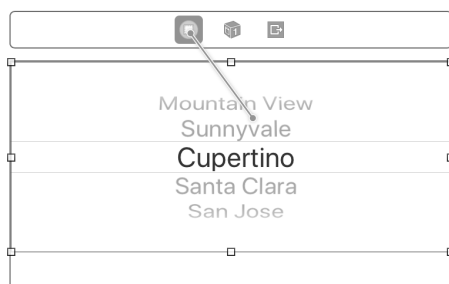
Picker View 預設只有一個滾輪，也透過屬性設定讓一個 Picker View 有兩個以上的滾輪，例如將出發地與目的地放在同一個 Picker View 上讓使用者選取。

Picker View 以三階段對話的方式來詢問 Picker View 上面要呈現什麼資料，三階段對話會以函數呼叫的方式來詢問，答案則是以 `return` 的方式回傳回去。除了 Picker View 之外，Table View 也會以三階段對話的方式來設定 Table 上要顯示的資料（請參考「表格」一章）。



步驟與說明

- 1 建立 Single View App 專案。
- 2 開啟 Main.storyboard，拖放一個 Picker View 到 View Controller 上。在 Picker View 上用滑鼠拉藍線拉到 View Controller 的黃圈圈上，勾選 `dataSource` 與 `delegate`。前者用來告訴 Picker View 要從哪個 view controller 中取得要顯示的資料，後者則是告訴 Picker View，當使用者選了選項後要讓哪一個 view controller 知道使用者的選擇。



3 開啟 ViewController.swift，先讓這個類別符合 UIPickerViewDataSource 與 UIPickerViewDelegate 這兩個協定的規範。然後再宣告兩個陣列，負責提供 Picker View 上兩個滾輪要呈現的資料。

```
class ViewController: UIViewController, UIPickerViewDataSource,
UIPickerViewDelegate {

    let fromList = ["台北松山", "桃園", "台中清泉崗", "高雄小港"]
    let toList = ["上海浦東", "香港赤臘角", "日本成田", "韓國仁川"]
```

4 第一階段對話：Picker View 詢問有幾個滾輪。

```
func numberOfComponents(in pickerView: UIPickerView) -> Int {
    return 2
}
```

5 第二階段對話：Picker View 詢問每個滾輪有幾筆資料。

```
func pickerView(_ pickerView: UIPickerView, numberOfRowsInComponent component:
Int) -> Int {
    // 0 代表最左邊的滾輪
    if component == 0 {
        return fromList.count
    }

    else if component == 1 {
        return toList.count
    }

    return 0
}
```

6 第三階段對話：Picker View 詢問每筆資料的實際內容為何。

```
func pickerView(_ pickerView: UIPickerView, titleForRow row: Int, forComponent
component: Int) -> String? {
    if component == 0 {
        return fromList[row]
    }

    else if component == 1{
        return toList[row]
    }

    return nil
}
```

7 三階段對話結束，Picker View 就可以正確的呈現資料了，但如果想要知道使用者到底選了什麼選項，就要實作 pickerView(_:didSelectRow:inComponent:) 函數。

```
func pickerView(_ pickerView: UIPickerView, didSelectRow row: Int, inComponent component: Int) {  
    if component == 0 {  
        print("出發地：\(fromList[row])")  
    }  
  
    else if component == 1 {  
        print("目的地：\(toList[row])")  
    }  
}
```

8 執行看看。



出發地：桃園

目的地：香港赤臘角

6-17

難易度



倒數計時器

先備知識：無

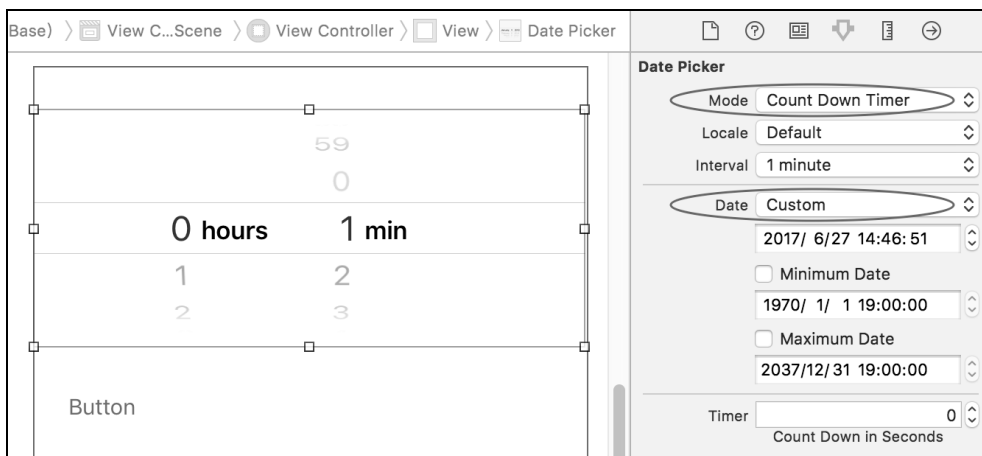
倒數計時器，Date Picker（類別為 UIDatePicker）。Date Picker 元件並不是只能用來讓使用者選取日期與時間，它還可以拿來設定倒數計時時間。



步驟與說明

1 建立 Single View App 專案。

2 開啟 Main.storyboard，拖放一個 Date Picker 元件到 View Controller 上。在屬性面板上將「Mode」改為 Count Down Timer，「Date」改為 Custom（若沒有改成 Custom，之後取得的倒數計時秒數不會是 60 秒的整數倍數）。然後再拖放一個 Button 元件用來取得使用者設定的倒數計時時間。



3 將 Xcode 畫面切換成輔助編輯模式，在 Date Picker 元件上拉藍線到 ViewController.swift 中，為 Date Picker 設定 IBOutlet 屬性。

```
class ViewController: UIViewController {  
  
    @IBOutlet weak var datePicker: UIDatePicker!
```

01

02

03

04

05

06

07

08

09

10

11

12

13

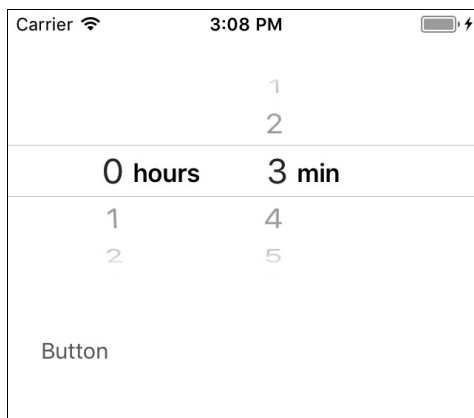
14

視覺化元件

4 在 Button 元件上用藍線拉出 Button 的 IBAction 函數。函數中透過 countdownDuration 屬性取得使用者設定的秒數。

```
@IBAction func onClick(_ sender: Any) {  
    let second = datePicker.countDownDuration  
    print(second)  
}
```

5 執行看看。



180.0

補充說明

透過 Date Picker 只能取得使用者設定的倒數計時秒數，如果要真的開始倒數，必須使用 Timer 類別來實作此部分，程式碼如下：

```
@IBAction func onClick(_ sender: Any) {  
    var second = datePicker.countDownDuration  
  
    Timer.scheduledTimer(withTimeInterval: 1.0, repeats: true) {  
        (timer) in  
            print("剩餘 \(second) 秒")  
            second = second - 1  
  
            if second < 0 {  
                timer.invalidate()  
            }  
    }  
}
```

網頁，Web View（類別為 WKWebView），此類別在 Xcode 中已具有視覺化元件，主要用來呈現網頁內容。一個設計良好的響應式網站不論是在桌上型電腦、行動裝置（iOS、Android）、或其他各種只要有瀏覽器的平台上都會有良好的顯示效果，此時在 App 端只要使用一個 Web View 元件去連結響應式網站的首頁就完成整個 App 設計了，不需要透過原生程式碼來分別開發不同平台的應用程式。

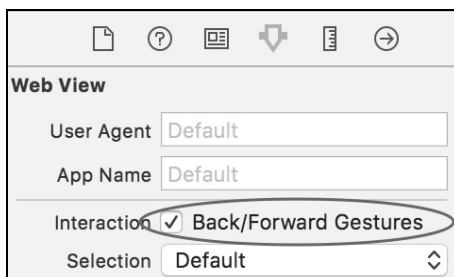
在 Xcode9 以前的 Web View 元件有兩個，一個是 UIWebView（有元件），另一個是 WKWebView（需全部使用程式碼來處理），從 Xcode9 開始，Web View 元件已經改成與 WKWebView 對應，而 UIWebView 已經被標上 deprecated 不建議使用了。

根據 Apple 的 App Transport Security（ATS）政策，當 Web View 元件要載入非 https 網址時必須關閉 ATS 或是例外排除，這部分請參考附錄 D。



步驟與說明

- 1** 建立 Single View App 專案。
- 2** 開啟 Main.storyboard，拖放一個 Web View 元件到 View Controller 上。
- 3** 在屬性面板上勾選「Back/Forward Gestures」，讓 Web View 支援手勢操作。



4 將 Xcode 畫面切換成輔助編輯模式，用藍線拉出 Web View 元件在 ViewController.swift 上的 IBOutlet 屬性，並且要匯入 WebKit 框架（如果 Xcode 出現找不到 WKWebView 類別的話）。

```
import WebKit

class ViewController: UIViewController {

    @IBOutlet weak var webView: WKWebView!
```

5 在 viewDidLoad() 中，讓 Web View 元件載入網址，這裡特別挑選了 http 協定的網址，因此必須在 Info.plist 檔中設定 ATS（請參考附錄 D）。

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

    let url = URL(string: "http://cnn.com")
    let request = URLRequest(url: url!)
    webView.load(request)
}
```

6 執行看看。

