24

iPhone 範例程式設計

假使您已對上述的章節了解後，就有能力來撰寫 iPhone 的程式。只要您有創意，又會寫 iPhone 程式，有一天您會從小卒變英雄。以下我們將以範例程式來解說一些基本的操作與程式的撰寫。由於本書主要談論的是 Objective-C 程式語言，所以沒有深入的加以論述，有興趣的讀者請參考有關 iPhone 程式設計的書籍。

24.1 範例程式一：按鈕互動之實作

本範例要介紹如何建立基本的 iPhone 應用程式，此程式使用按鈕來與使用者互動，並使用 Label 顯示使用者按下哪個按鈕。圖 24-1 為本範例程式的執行結果之畫面：

圖 24-1　iPhone 模擬器上程式執行畫面

首先，點選 Xcode 系統列上的「File→New Project…」開啟新專案，如圖 24-2。請在左欄中選取 iPhone OS 下的 Application，在右上方欄中點選 View-based Application，此時 Xcode 會自動幫你建立一些程式樣板，包括介面檔案 xib，以及可以撰寫程式控制的 view controller。

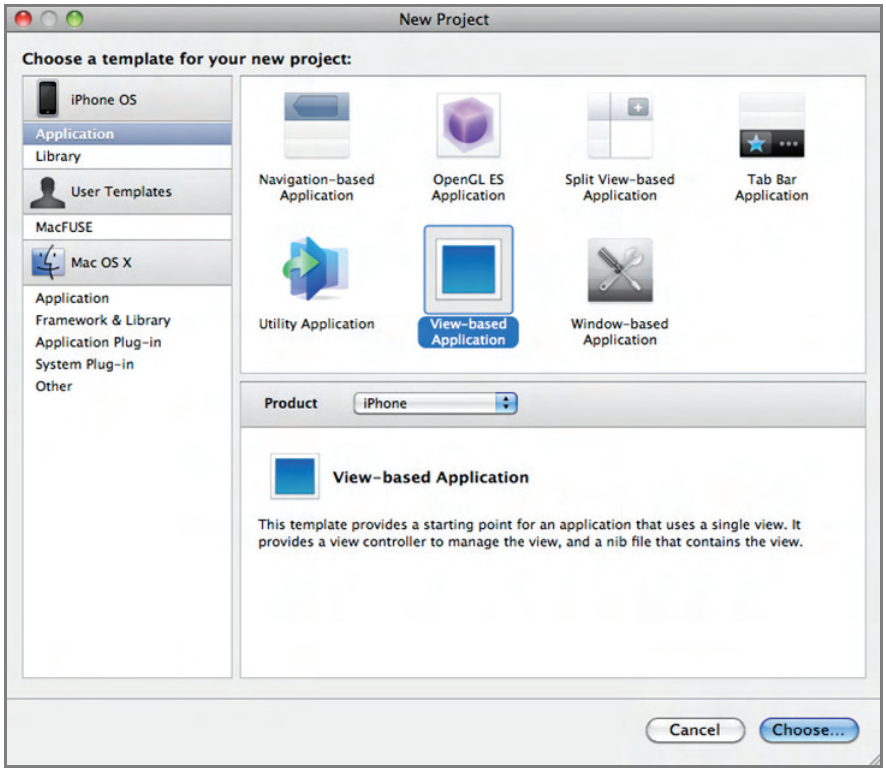

圖 24-2　iPhone 專案選擇視窗

按下「Choose…」後，輸入專案名稱 “buttonExample”，儲存並開啟專案。專案開啟後，將會看見如圖 24-3 的程式編輯畫面：

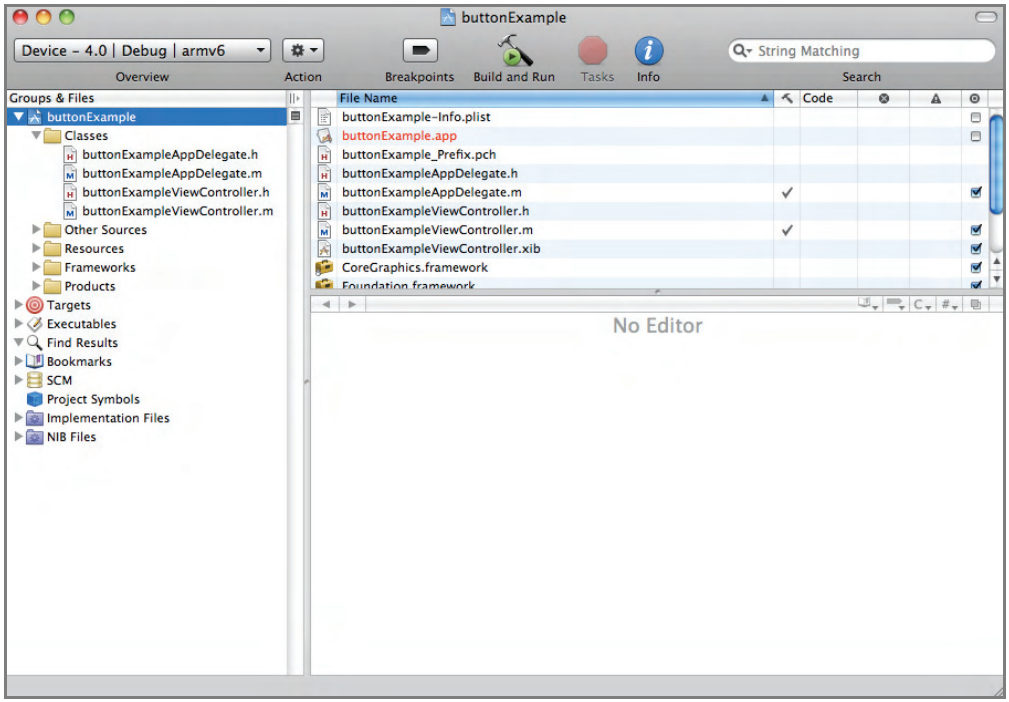

圖 24-3　程式編輯畫面

現在開始撰寫程式碼。首先，選擇 buttonExampleViewController.h，輸入以下程式碼：

程式名稱：buttonExampleViewController.h

```
// buttonExampleViewController.h
#import <UIKit/UIKit.h>
@interface buttonExampleViewController : UIViewController {
    IBOutlet UILabel *myLabel;
}
@property (retain, nonatomic) UILabel *myLabel;

-(IBAction) tapA:(id)sender;
-(IBAction) tapB:(id)sender;

@end
```

編輯完成後，按下「File→Save」儲存檔案。接下來，開始製作我們 iPhone 應用程式的介面！

點選 buttonExampleViewController.xib 檔案，將會開啟 Interface Builder。開啟後將會看見以下兩個視窗，其中，圖 24-4 是 Interface Builder 物件視窗，而圖 24-5 為應用程式的 View 視窗。

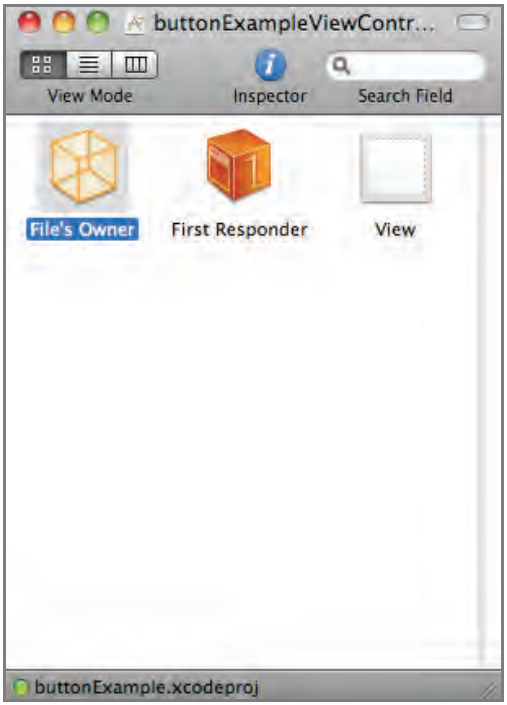

圖 24-4　Interface Builder 物件視窗

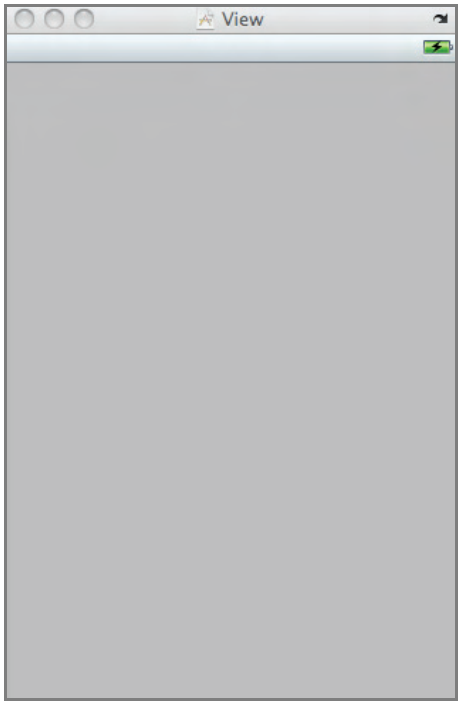

圖 24-5　View 視窗

在此，我們建議點選 Interface Builder 系統列上的「tools」，並點選「inspector」與「library」開啟檢視器及 Library。開啟後會看見如圖 24-6 兩個視窗。在 Library 中，提供了許多物件，供你拖曳至 View 中，如圖 24-6 右邊所示。Inspector 可以調整物件的屬性，如在 Library 視窗中選擇 Label，此時就可以在 View Attribute 視窗中調整 Label 的標題、背景、字型…等等，如圖 24-6 左邊所示。

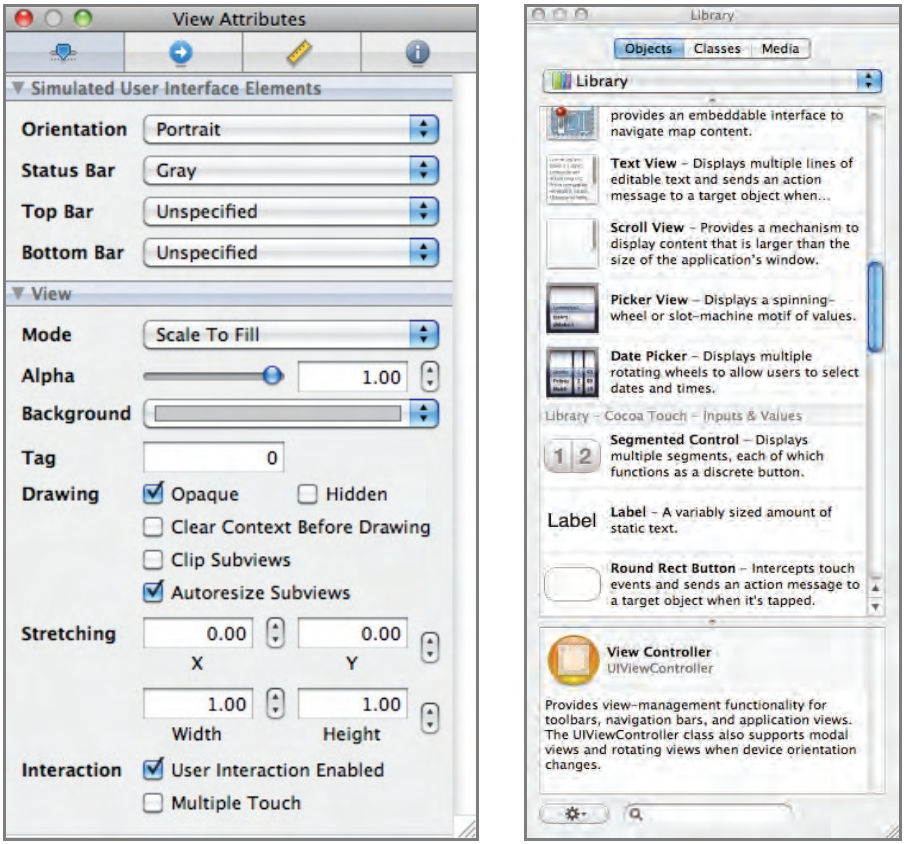

圖 24-6 Inspector 與 Library 視窗

現在，我們可以從 Library 中拖曳一個 Label 與兩個 Button 至 View 視窗，並用左鍵雙觸擊 Button 以更改按鈕 title，其名稱分別為 A 與 B，完成後的介面如圖 24-7 所示。

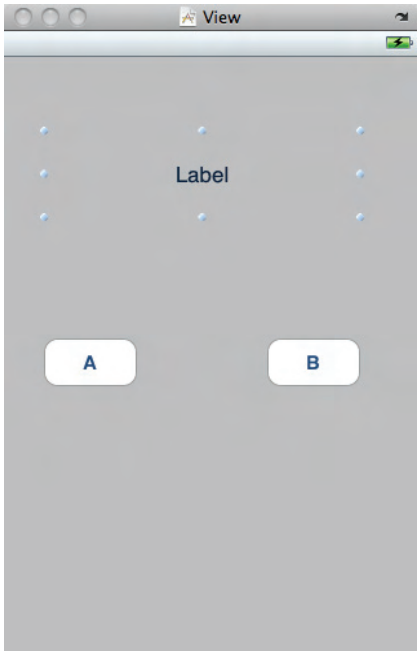

圖 24-7　View 視窗編輯畫面

接下來，將 Label 與 Button 和先前所宣告的程式碼連結。如圖 24-8 所示。

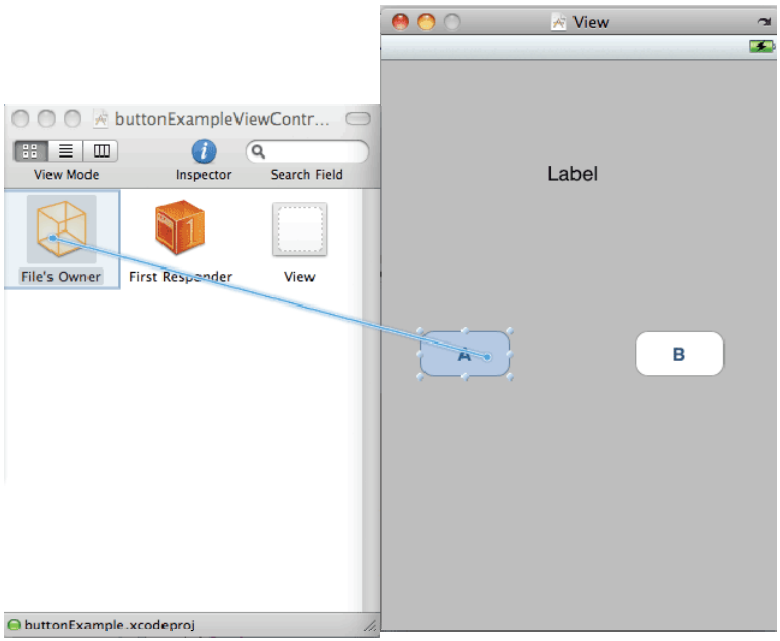

圖 24-8　物件連結

按住 control 鍵，並在 A button 上按左鍵拖曳至「file's owner」，拖曳時將會看見一條連結線，將這條線從 A button 連結至 file's owner，並放開左鍵，此時會看見我們之前預先宣告的 IBAction 方法，請點選 tapA: 方法即完成連結，B button 也是以同樣的作法連結 tapB: 方法，如圖 24-9 所示。

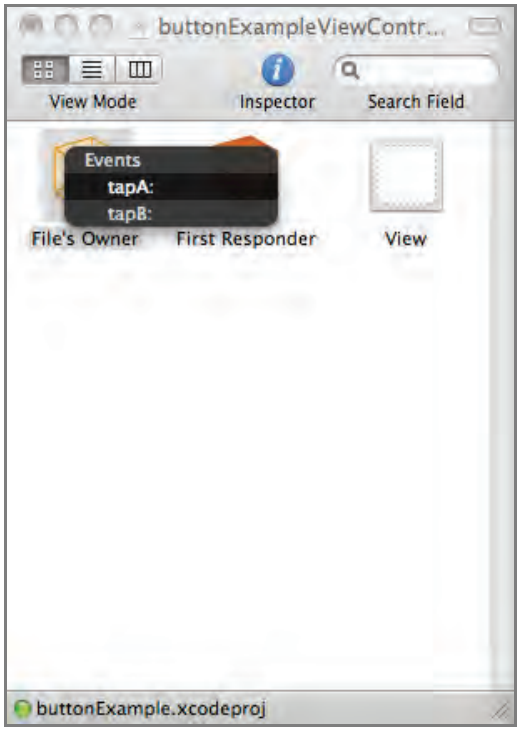

圖 24-9 設定按鈕與方法的連結

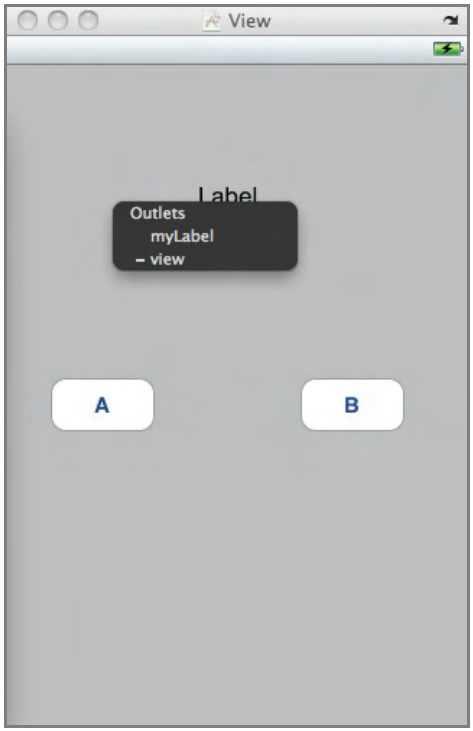

圖 24-10 設定標籤與方法的連結

同樣的，按著 control 與滑鼠左鍵並將「file's owner」連線到 Label 上，也可以看見先前所宣告的 myLabel，如圖 24-10 所示。由於我們要將結果輸出於 Label，所以和上述的動作是相反的方向。

現在，View 上的物件都已經連接完成，可以開始實作之前所定義的方法了。

打開 buttonExampleController.m，並輸入以下程式碼：

程式名稱：buttonExampleController.m

```
//buttonExampleController.h
#import "buttonExampleViewController.h"
@implementation buttonExampleViewController
@synthesize myLabel;
-(void) tapA:(id)sender {
    myLabel.text = @"我按了 A 按鈕";
}
-(void) tapB:(id)sender {
    myLabel.text = @"我按了 B 按鈕";
}
@end
```

輸入完成後，按下 Build and Run 即可開始編譯並執行程式。

執行的結果如圖 24-1 所示。

24.2 範例程式二：圖片與按鈕互動實作

本範例要介紹如何使用 UIImageView 在程式中加入圖片，並藉由按鈕切換圖片。在程式中加入圖片可以使你的程式更加活潑，使得介面更加美觀。UIimageView 可以支援許多格式的圖片檔案，如 jpg、gif、png 等常見格式。圖 24-11 為本程式執行畫面。

圖 24-11　程式執行的畫面

開始撰寫程式之前，必須先把要顯示於程式中的圖片放入專案中，以便之後程式讀取使用。將圖片拖曳進專案所在的資料夾中，之後再將圖片拖曳進入 Xcode 程式編輯視窗中左側的 Resource 資料夾，如圖 24-12 所示。

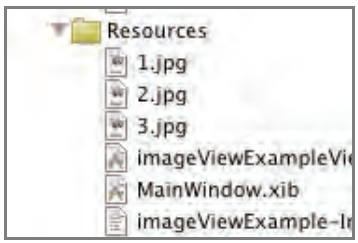

圖 24-12 將圖片檔加入於 Resource 資料夾

現在開始撰寫我們的程式。首先打開一個新的 View-Based 專案並命名為 imageViewExample。接著，開啟 imageViewExampleViewController.h 填入以下程式碼：

程式名稱：imageViewExampleViewController.h

```
// imageViewExampleViewController.h
#import <UIKit/UIKit.h>
@interface imageViewExampleViewController : UIViewController {
    IBOutlet UIImageView *imageView;
}
-(IBAction) pic1ButtonTapped;
-(IBAction) pic2ButtonTapped;
-(IBAction) pic3ButtonTapped;
@property (retain, nonatomic) UIImageView *imageView;
@end
```

此處定義了一個 UIImageView 與三個處理按鈕動作的方法。接著實作以上所定義的方法。

程式名稱：imageViewExampleViewController.m

```
//imageViewExampleViewController.m
#import "imageViewExampleViewController.h"
@implementation imageViewExampleViewController
@synthesize imageView;
-(IBAction)pic1ButtonTapped {
    imageView.image = [UIImage imageNamed:@"1.jpg"];
}
-(IBAction)pic2ButtonTapped {
    imageView.image = [UIImage imageNamed:@"2.jpg"];
}
```

```
-(IBAction)pic3ButtonTapped {
    imageView.image = [UIImage imageNamed:@"3.jpg"];
}
- (void)dealloc {
    [imageView release];
    [super dealloc];
}
```

實作三個按鈕行為的方法中，分別將圖片 1、2、3 傳給 UIImageView，使程式顯示傳入的圖片。

24.3 範例程式三：文字輸入與顯示之實作

本範例示範了如何利用文字編輯框，並且將使用者所輸入的文字顯示於 Label 上。文字編輯框是讓使用者可以自由輸入文字的介面，在許多軟體中都很常見。輸出的畫面如圖 24-13 所示：

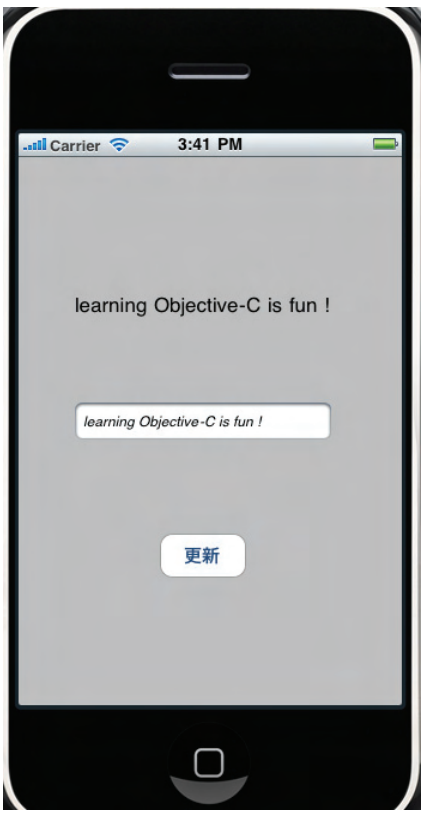

圖 24-13　程式執行的畫面

和範例程式一之實作相同，首先，開啟一個 View-based Application 專案，並命名為 editTextExample。打開 editTextExample.h，並加以編輯此標頭檔。

程式名稱：editTextExampleViewController.h

```
#import <UIKit/UIKit.h>
@interface editTextExampleViewController : UIViewController {
    IBOutlet UILabel *display;
    IBOutlet UITextField *textField;
}
@property (retain,nonatomic) IBOutlet UILabel *display;
@property (retain,nonatomic) IBOutlet UITextField *textField;
- (IBAction) updateLabel:(id)sender;
- (IBAction) doneEditing:(id)sender;
@end
```

此處宣告了兩個方法，updateLabel 方法是當按下圖 24-13 的更新按鈕時，文字編輯框內的文字會傳送到 Label 上。而 doneEditing 方法是按下虛擬鍵盤上的 “Return” 時，表示完成編輯，並將虛擬鍵盤關閉。接著實作以上定義的方法：

程式名稱：editTextExampleViewController.m

```
#import "editTextExampleViewController.h"
@implementation editTextExampleViewController
@synthesize display,textField;

-(IBAction) doneEditing:(id)sender {
    [sender resignFirstResponder];
}

-(IBAction) updateLabel:(id)sender {
    display.text = textField.text;
}
```

在 doneEditing 方法中，呼叫 resignFirstResponder，這個方法可以讓物件回到初始狀態，因此，可以讓虛擬鍵盤關閉。關於 resignFirstResponder 的細節，可以參考 UIResponder 的說明文件，由於 TextField 類別有繼承 UIResponder 類別，故此處我們可以直接呼叫此方法。

updateLabel 方法中，我們僅單純的將 textField 的文字，以更新與顯示 Label 上文字。

實作完成後，便可開啟 editTextExampleViewController.xib 檔案編輯介面。

首先，拖曳出 Label、TextField，以及 Button 各一個至 View 視窗中，並加以更改 Button 的 title 名稱，使得初始畫面如圖 24-14 所示：

圖 24-14　初始畫面

再來將 Label 與 File's Owner 連線。完成 Label 的連線後，接著要將文字編輯框連線，不過此處連線的方法與之前不同，我們需要特別指定文字編輯框完成編輯時的動作。因此連線時，需先點選 TextField，打開 Inspector，選擇第二個標籤，將 Did End On Exit 與 File's owner 中的 doneEditing 方法連線，如圖 24-15 所示：

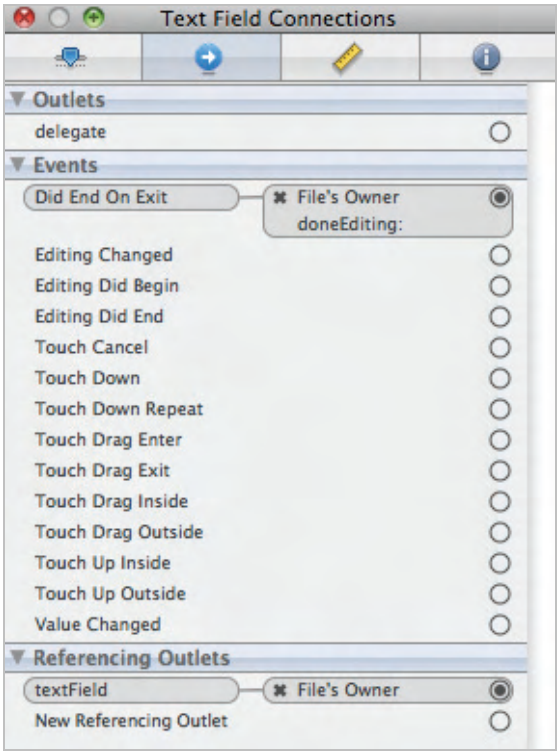

圖 24-15 Text Field 相關訊息的設定

連線完成後，即可開始執行程式，其結果如圖 24-13 所示。

24.4 範例程式四：開關與滑桿之實作

本範例要介紹在 iPhone 中兩個基本的 UI 控制項目，分別是 Switch 與 Slider。Switch 通常用於控制狀態的開關，你可以發現在 iPhone 的設定畫面使用了大量的 Switch。Slider 是一條滑桿，通常控制著音量、亮度。圖 24-16 控制自動亮度是否開啟的開關即是 Switch，而調整亮度的滑桿即為 Slider。

圖 24-16　iPhone 亮度控制設定畫面

以下示範如何撰寫程式碼，以實作 Switch 與 Slider。這個程式使用兩個 Label 分別顯示 Switch 與 Slider 當前狀態。我們預定的完成畫面如圖 24-17：

圖 24-17　程式完成畫面

現在開始示範如何完成這個範例。首先，開啟一個 View-Base Application 專案，命名為 **SliderSwitchExample**。接著打開 **SliderSwichExampleViewController.h** 進行編輯，輸入以下程式碼：

程式名稱：SliderSwitchExampleViewController.h

```
#import <UIKit/UIKit.h>
@interface SliderSwitchExampleViewController : UIViewController {
    IBOutlet UILabel *switchState;
    IBOutlet UILabel *sliderState;
}
@property (retain, nonatomic) IBOutlet UILabel *switchState;
@property (retain, nonatomic) IBOutlet UILabel *sliderState;
-(IBAction) switchChange:(id)sender;
-(IBAction) sliderChange:(id)sender;
@end
```

這裡宣告了兩個 Label 及兩個方法，這兩個方法分別是控制當 Switch 或 Slider 改變時所呈現的動作。完成後打開 SliderSwichExampleViewController.m，開始實作這兩個方法。

程式名稱：SliderSwitchExampleViewController.m

```
#import "SliderSwitchExampleViewController.h"
@implementation SliderSwitchExampleViewController
@synthesize switchState,sliderState;
-(IBAction) switchChange:(id)sender {
    UISwitch *mySwitch = (UISwitch *)sender;
    if (mySwitch.on) {
        switchState.text = @"Switch is ON!";
    }else {
        switchState.text = @"Switch is OFF!";
    }
}
-(IBAction) sliderChange:(id)sender {
    UISlider *mySlider = (UISlider *)sender;
    int number = (int)(mySlider.value+0.5f);
    NSString *state = [[NSString alloc] initWithFormat:@"%d",number];
    sliderState.text = state;
    [state release];
}
- (void)dealloc {
    [super dealloc];
}
```

在 switchChange 方法中，先宣告一個 UISwitch 接受從參數 sender 傳送進來的 Switch。在 if 的判別敘述中，mySwitch.on 會傳回一個 Bool 值，表示 Switch 的狀

態為 On 或 Off，若為 On 則為 True，Off 為 False。當 Switch 是 ON 的時候，顯示 Switch 狀態的 Label 的文字就為 Switch is ON!，反之，則為 Switch is OFF!。

在 sliderChange 的方法中，同樣宣告一個 UISlider 來接收從參數 sender 傳入的 Slider，而當 Slider 有所變化時，就會將 Slider 的數值轉為整數 int 的型態，再將它變為字串，傳遞給顯示 Slider 狀態的 Label。

如此一來，程式撰寫的部份已經完成。接著我們要來建立介面的部份。

打開 SliderSwitchExampleViewController.xib 編輯介面，從 Library 中拖曳出兩個 Label 做為 Slider 和 Switch 的顯示狀態，以及 Slider 和 Switch，另外還要再拖曳出兩個 Label 做為 Slider 最大值與最小值的標示，並把它們如圖 24-17 那般排列。

接著，點擊 Slider，並打開 inspector 來設定其最大值以及最小值，使 Slider 的控制範圍在 0 到 100 之間，並且將初始值（initial）設定為 50，如圖 24-18 所示：

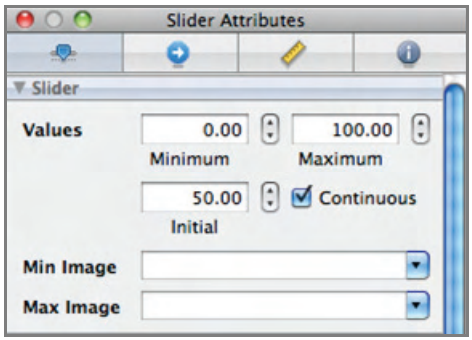

圖 24-18　Slider 控制元件相關的設定

將 Switch 狀態的 Label 與 Slider 狀態的 Label 和 File's owner 連結起來，再將 Switch 與 changeSwitch 方法連結、Slider 與 changeSlider 方法連結，即可開始執行與測試。

詳細的連結方法在按鈕的範例中已經詳述了，在此不再贅述。

圖 24-19 為 Switch 為 ON，而且 Slider 之值為 90 時的行畫面。