畫面元件

王昱景　Brian Wang
brian.wang.frontline@gmail.com

一切畫面元件的開始 UIView

- UIView 會佔有一塊矩形區域，是以 CGRect 這個 C 結構來表示
- UI View 的一切描繪動作以及事件處理都發生在這個區域
- 可描繪其他的東西，也可以放入或移除元件
- 繼承了 UIResponder 類別，定義了包括觸碰事件及搖晃事件等

- 一個 UIView 底下可以擁有多個子 UIView，後加入的 UIView會被放置在上層
- 可以透過調整這些 UIView 的上下層關係來構成畫面的轉換
- 透過加入手勢的識別，可讓 UIView 識別出使用者的動作並做出相對應的回應
- 用區域的原點 (origin) 及其大小 (size) 定義，圓點包括 X 座標與 Y 座標，尺寸包括了寬度 (width) 與高度 (height)

- CGPoint - 在畫面座標上的一點

```
struct CGPoint {
    CGFloat x;
    CGFloat y;
};
typedef struct CGPoint CGPoint;
```

- CGSize - 在畫面上的一塊區域大小

```
struct CGSize {
    CGFloat width;
    CGFloat height;
};
typedef struct CGSize CGSize;
```

- CGRect - 在畫面上一塊區域的位置與大小

```
struct CGRect {
    CGPoint origin;
    CGSize size;
};
typedef CGRect CGRect;
```

- CGPointMake(x, y)：用來產生 CGPoint

```
CGPoint point = CGPointMake(100.0, 200.0);
```

- CGSizeMake(width, height)：用來產生 CGSize

```
CGSize size = CGSizeMake(40.0, 50.0);
```

- CGRectMake(x, y, width, height)：用來產生 CGRect
  CGRect rect = CGRectMake(0, 0, 320, 240);

- CGSizeEqualToSize(size1, size2)：用來比較兩個 CGSize 是否相同
  CGSize size1 = CGSizeMake(100, 200);
  CGSize size2 = CGSizeMake(200, 100);
  BOOL isEqual = CGSizeEqualToSize(size1, size2);

- CGRectContainsPoint(rect, point)：用來檢查變數 point 是否在 rect 所包含的區域內
  CGPoint point = CGPointMake(10, 10);
  CGRect rect = CGRectMake(5, 5, 30, 30);
  BOOL isContainsPoint =
  CGRectContainsPoint(rect, point);

- CGRectGetMinX(rect) 和 CGRectGetMinY(rect)：分別取得區域 rect 左上角的 X, Y 座標

- CGRectGetMidX(rect) 和 CGRectGetMidY(rect)
  ：分別取得區域 rect 中心點的 X, Y 座標

- CGRectGetMaxX(rect) 和 CGRectGetMaxY(rect)
  ：分別取得區域 rect 右下角的 X, Y 座標

- 定義一個 UIView
  UIView *view = [[UIView alloc]
  initWithFrame:CGRectMake(0, 0, 320, 480)];

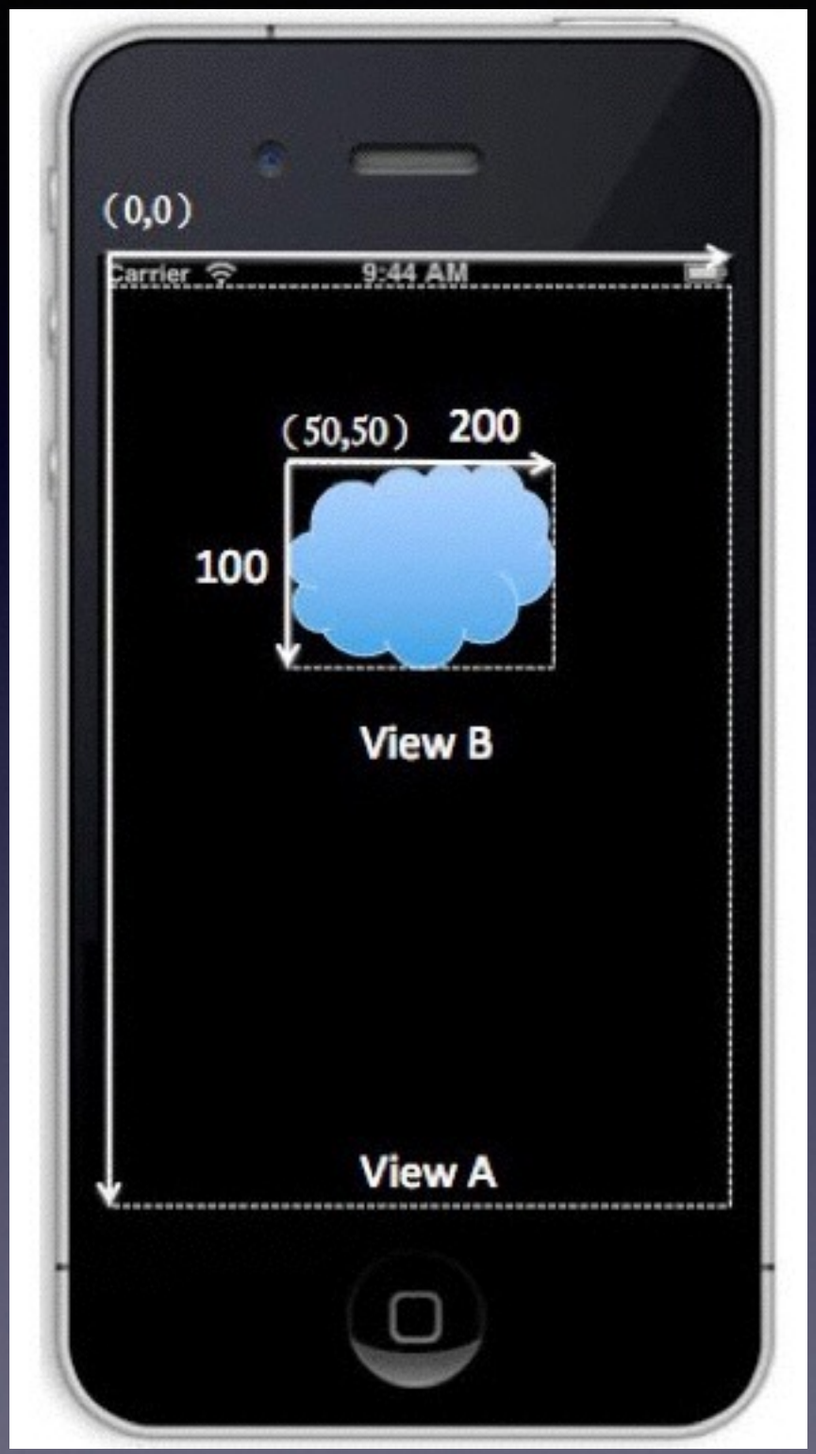
- CGRectNull：CGRect 空值常數，使用 CGRectIsNull() 函數來判斷

- CGRectZero：定義出一個大小為 0 的區域

```
UITableViewCell *cell = [[UITableViewCell alloc]
initWithFrame:CGRectZero reuseIdentifier:MyId];
```

- CGRectInfinite：產生一個無窮大的區域

- 座標系統



- setFrame:(CGRect *)rect - 相對座標
  CGRect *rect = CGRectMake(50, 50, 200, 100);
  [viewB setFrame:rect]; // 或viewB.frame = rect;

- setBounds:(CGRect *)rect - 絕對座標
  [viewB setBounds:CGRectMake(0, 0, 200, 100)];

- CoreGraphics 框架庫採用標準的迪卡爾座標系統 (Cartesian coordinates)，原點由左下角算起

- 如果需要使用到 CoreGraphics 來繪製圖形到 iOS 上，必須在 UIView 子類別的 drawRect: 方法做座標轉換

```objc
-(void)drawRect:(CGRect)rect {
    // 取得現在的 context 物件
    CGContextRef context = UIGraphicsGetCurrentContext();

    // 轉換座標
    CGContextTranslateCTM(context, 0.0, rect.size.height);
    CGContextScaleCTM(context, 1.0, -1.0);

    // 用 CGContextDrawImage 等函數繪圖在 iOS 上
}
```

- 子畫面的處理
  - 加入一個 UIView
    - (void)addSubview:(UIView *)view;
  - 加入後的 UIView 會一層層的覆蓋在父畫面上

```
UIView *parentView = [[UIView alloc]
initWithFrame:frame];
UIView *viewA = [[UIView alloc] initWithFrame:frame];
UIView *viewB = [[UIView alloc] initWithFrame:frame];
[parentView addSubview:viewA];
[parentView addSubview:viewB];
```

- 將本身自上一層的 UIView 內移除
  - (void)removeFromSuperview;

- 交換兩個 UIView 的順序
  - (void)exchangeSubviewAtIndex:(int)index withSubviewAtIndex:(int)otherIndex;

  [parentView exchangeSubviewAtIndex:0 withSubviewAtIndex:1];

- 將某個 UIView 帶到前面
  - (void)bringSubviewToFront:(UIView *)view;

  [parentView bringSubviewToFront:viewA];

- UIView 的標籤
  viewA.tag = 100;
  viewB.tag = 101;
  UIView *view = [parentView viewWithTag:100];

- 重繪與多點觸碰事件處理
  - UIView 除了負責畫面的處理外，也繼承了 UIResponder 類別，所以也必須處理畫面的重繪以及多點觸碰事件
  - 重繪事件的處理
    - - (void)drawRect:(CGRect *)rect;
    - 動態繪製某些圖形到 UIView 時需要重繪
    - 強迫 iOS 重繪不要呼叫，而是透過 UIView 的 setNeedsDisplay 來更新畫面，或是 setNeedsDisplayInRect: 局部更新某個區域畫面

- 多點觸碰事件處理
  - - (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event;
  - - (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event;
  - - (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event;

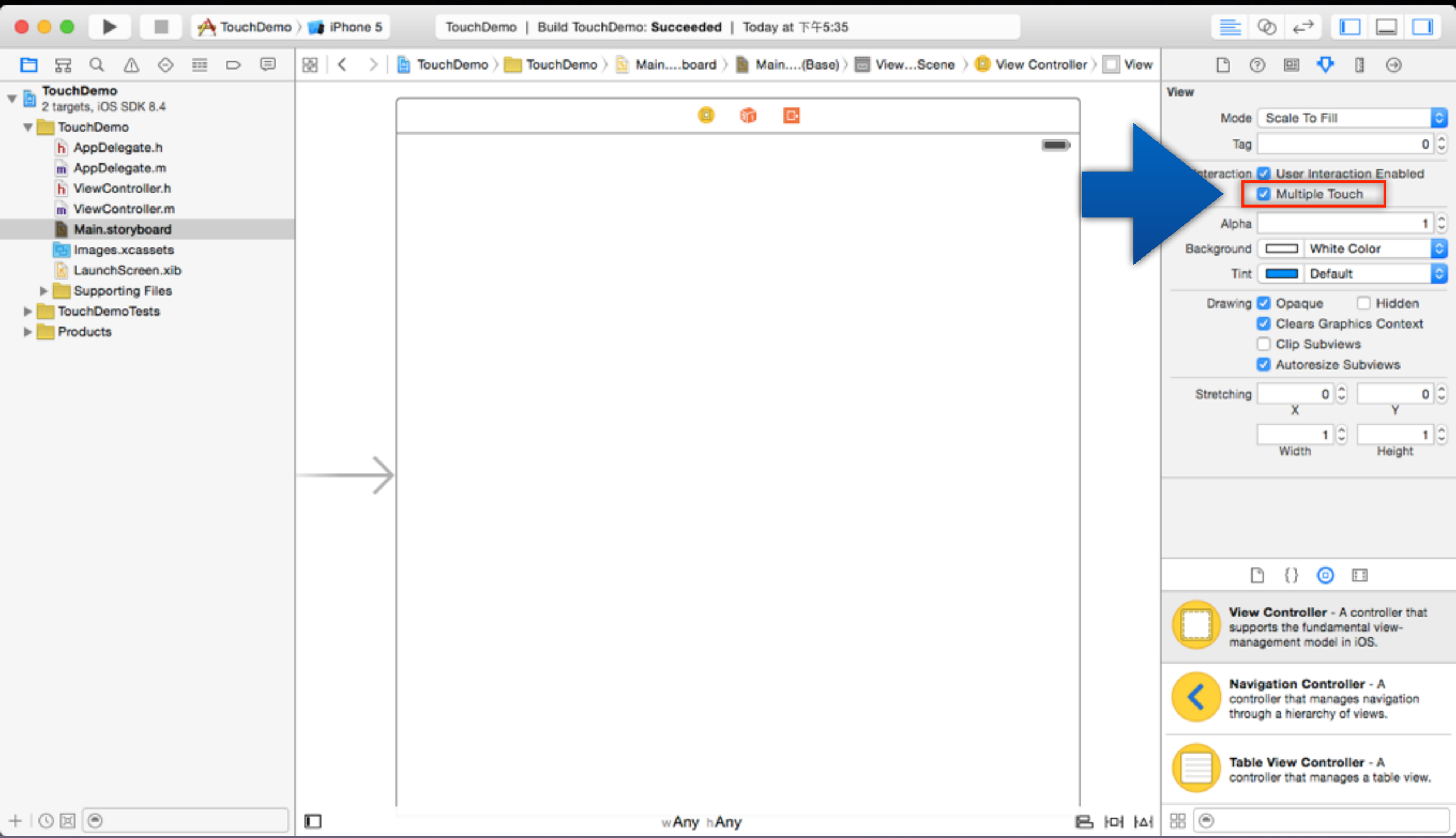
- 觸碰到 UIView 所屬的區域時自動被呼叫
- 碰到螢幕一開始就會發送一次 touchesBegan:withEvent: 的訊息
- touches 代表不分先後順序的多個觸碰點
- event 為觸碰的事件，可取得觸碰事件內有幾個輕拍數 (tapCount)
- 當按住畫面移動時 touchesMoved:withEvent: 的訊息則會被發送出去，一直到放開手指後才會發送 touchesEnded:withEvent: 訊息

```objc
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
    UITouch *touch = [touches anyObject];
//    UITouch *touch = [[touches allObjects] objectAtIndex:0];

    CGPoint location = [touch locationInView:self]; // 取得觸碰點

    if ([touch tapCount] == 2) { // 若輕拍兩下

    }
}
```

- 一般情況下並不在意使用者使用了幾根手指，而只在乎使用者的輕拍動作
- 透過 NSSet 裡面的 anyObject 方法可以取得任一個觸碰點
- 透過 allObjects 可以取得所有的觸碰點並自己一一處理這些觸碰點
- 還可以夠過 UITouch 類別的 locationInView: 來取得觸碰點位置，以及 tapCount 來取得使用者輕拍的次數

```objc
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController {
    CGFloat redColor;
    CGFloat greenColor;
    CGFloat blueColor;
}

@end
```

```objc
- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    int count = [touches count];

    switch(count) {
        case 3:
            blueColor = (blueColor > 1) ? 0 : blueColor + 0.05;
        case 2:
            greenColor = (greenColor > 1) ? 0 : greenColor + 0.05;
        case 1:
            redColor = (redColor > 1) ? 0 : redColor + 0.05;
        default:
            break;
    }

    self.view.backgroundColor = [UIColor colorWithRed:redColor green:greenColor blue:
        blueColor alpha:1.0];
}
```


TouchDemo 〉 iPhone 5
TouchDemo | Build TouchDemo: Succeeded | Today at 下午5:35
TouchDemo 〉 TouchDemo 〉 Main....board 〉 Main....(Base) 〉 View...Scene 〉 View Controller 〉 View
TouchDemo
2 targets, iOS SDK 8.4
TouchDemo
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Images.xcassets
LaunchScreen.xib
Supporting Files
TouchDemoTests
Products
View
Mode
Scale To Fill
Tag
0
User Interaction Enabled
Multiple Touch
Alpha
1
Background
White Color
Tint
Default
Drawing
Opaque
Hidden
Clears Graphics Context
Clip Subviews
Autoresize Subviews
Stretching
0
0
X
Y
1
1
Width
Height
View Controller - A controller that supports the fundamental view-management model in iOS.
Navigation Controller - A controller that manages navigation through a hierarchy of views.
Table View Controller - A controller that manages a table view.
wAny hAny

- 具有圓角的 UIView
  - 將 CoreGraphics.framework 加到專案中
  - 引用 QuartzCore.h
- 改變圓角的半徑

```
view.layer.masksToBounds = YES;
view.layer.cornerRadius = 11;
```

- 動畫效果
  - 簡單的動畫效果可以透過 UIImageView 與 UIView 兩個類別實現
- 用 UIImageView 做出動畫效果
  - UIImageView 允許透過播放數張靜態圖檔的方式來達到動畫效果 (類似動態 GIF 圖檔)

```objc
// 先把要播放的畫面順序排好
NSMutableArray *anims = [[NSMutableArray alloc] init];
UIImage *img1 = [UIImage imageNamed:@"pic1.png"];
UIImage *img2 = [UIImage imageNamed:@"pic2.png"];
[anims addObject:img1];
[anims addObject:img2];

// 再用 UIImageView 播放畫面
UIImageView *imageView = [[UIImageView alloc] initWithFrame:CGRectMake(0.0f, 0.0f,
    40.0f, 40.0f)];
imageView.animationImages = anims;
imageView.animationDuration = 1.0f; // 播放一次動畫所需時間
imageView.animationRepeatCount = 0; //一直播放
[imageView startAnimating];
```

- UIView 內建的動畫效果
  - 兩個畫面切換時或是在同一個畫面產生變化的動畫效果
  - 最常使用的一種方式
  - 把畫面變化後的程式碼夾在 beginAnimations:context: 與 commitAnimations 之間
  - 然後給予一個動畫時間以及動畫的時間曲線
  - iOS 會自動計算出變化前與變化後的正確路徑

```objc
// 畫面變化前的程式碼
// ...

[UIView beginAnimations:@"myAnim" context:NULL]; // 動畫開始
[UIView setAnimationCurve:UIViewAnimationCurveEaseInOut]; // 動畫時間曲線
[UIView setAnimationDuration:0.5f]; // 動畫時間

// 畫面變化後的程式碼
// ...

[UIView commitAnimations]; // 動畫結束
```

- 動畫開始
  - + (void)beginAnimations:(NSString *)animationId context:(void *)context;
    - 用來宣告一個動畫的開始
    - animationId 是應用程式內所設定的一個 Id，可以任意命名用以識別這個動畫區塊

- 動畫時間曲線
- + (void)setAnimationCurve:(UIViewAnimationCurve)curve;
  - 用來定義動畫播放的時間曲線
    - UIViewAnimationCurveLinear：整個動畫過程的播放速度保持一致
    - UIViewAnimationCurveEaseIn：在動畫開始時慢慢加速，到快結束時則保持一定速度，形成 S 曲線的前半段

- UIViewAnimationCurveEaseOut：在動畫開始時速度很快加速，然後再慢慢減速形成 S 曲線的後半段
- UIViewAnimationCurveEaseInOut：在動畫開始時慢慢加速，在結束前又慢慢減速，構成一個完整的 S 曲線，最常用

- 動畫播放時間
  - + (void)setAnimationDuration: (double)delay;
  - 定義動畫播放時間
    - 以秒為單位，預設為 0.25 秒
- 動畫結束
  - + (void)commitAnimations;
  - 整個動畫區塊的結束

```objc
// 淡出效果
self.view.alpha = 1.0f; // 先設定畫面 100% 顯示出來
[UIView beginAnimations:@"myAnim" context:NULL]; // 動畫開始
[UIView setAnimationCurve:UIViewAnimationCurveEaseInOut]; // 動畫時間曲線
[UIView setAnimationDuration:0.5f]; // 動畫時間
self.view.alpha = 0.0f; // 讓畫面消失
[UIView commitAnimations]; // 動畫結束
```

```objc
// 變形效果
self.view.transform = CGAffineTransformScale(self.view.transform, 1.0f, 1.0f); // 設定縮放比例為 1
[UIView beginAnimations:@"myAnim" context:NULL]; // 動畫開始
[UIView setAnimationCurve:UIViewAnimationCurveEaseInOut]; // 動畫時間曲線
[UIView setAnimationDuration:0.5f]; // 動畫時間
self.view.transform = CGAffineTransformScale(self.view.transform, 2.0f, 2.0f); // 設定縮放比例為 1
[UIView commitAnimations]; // 動畫結束
```

```objc
// 轉場 (Transition) 效果
[UIView setAnimationTransition:UIViewAnimationOptionTransitionFlipFromLeft forView:
    self cache:YES]; // 翻轉效果
[UIView beginAnimations:@"myAnim" context:NULL]; // 動畫開始
[UIView setAnimationCurve:UIViewAnimationCurveEaseInOut]; // 動畫時間曲線
[UIView setAnimationDuration:0.5f]; // 動畫時間
[self.view exchangeSubviewAtIndex:0 withSubviewAtIndex:1]; // 交換兩個畫面
[UIView commitAnimations]; // 動畫結束
```

UIViewAnimationOptionTransitionFlipFromRight：從右邊翻轉到左邊
UIViewAnimationOptionTransitionFlipFromLeft：從左邊翻轉到右邊
UIViewAnimationOptionTransitionCurlUp：往上翻頁
UIViewAnimationOptionTransitionCurlDown：往下翻頁
UIViewAnimationOptionTransitionNone：無任何效果

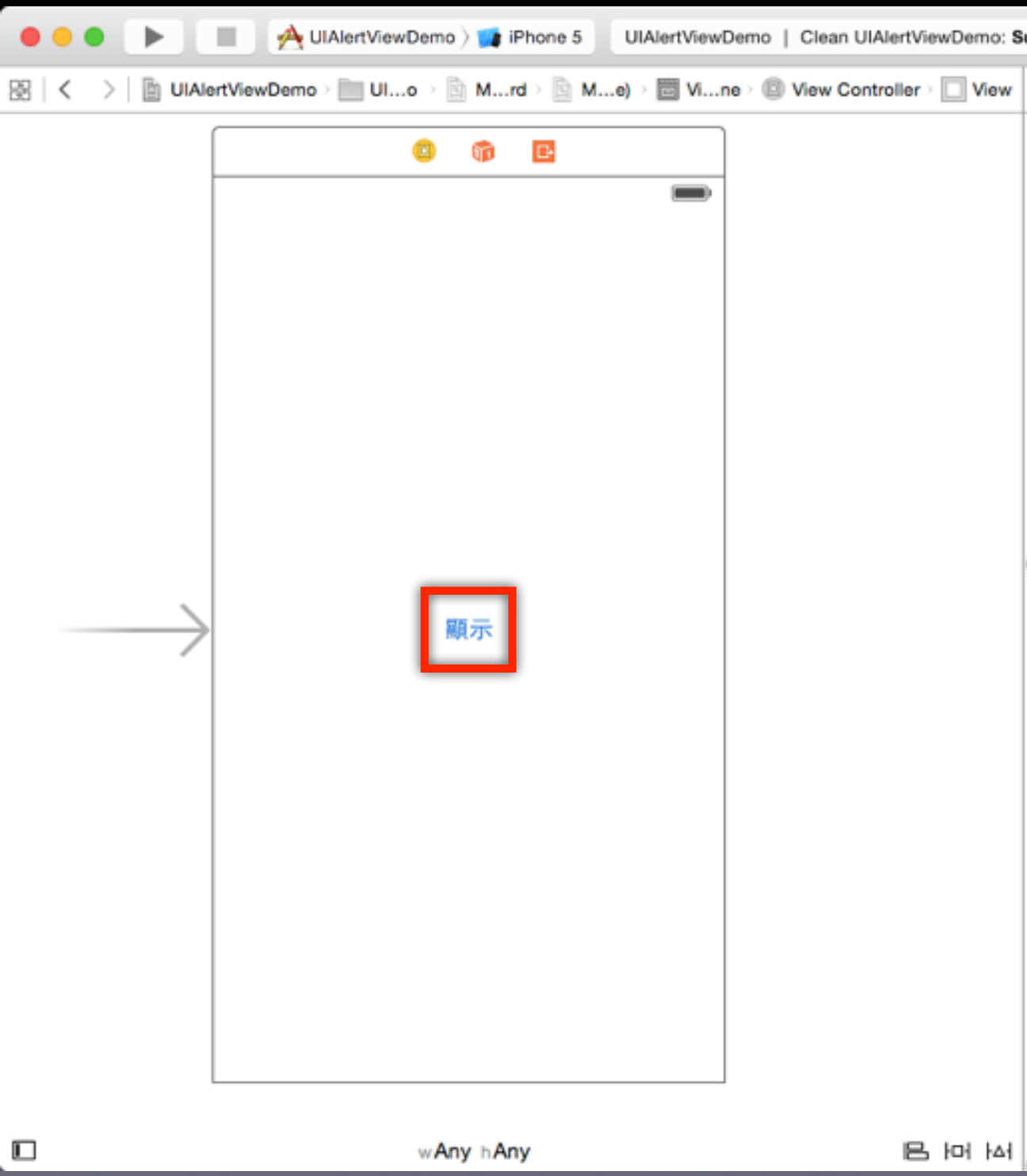
```objc
// Block 用法
[UIView animateWithDuration:0.5f
                      delay:0.1f
                    options:(UIViewAnimationOptionTransitionFlipFromLeft|
                             UIViewAnimationOptionCurveEaseInOut)
                 animations:^{
                     [self.view exchangeSubviewAtIndex:0 withSubviewAtIndex:1];
                 }
                 completion:^(BOOL finished){
                     if (finished) {
                         NSLog(@"Done!");
                     }
                 }
];
```

```objc
// 只針對某一個 UIView 進行轉場效果
self.view.alpha = 0;
[UIView transitionWithView:self.view
                  duration:0.5f
                   options:(UIViewAnimationOptionTransitionCurlUp)
                animations:^{
                    self.view.alpha = 1;
                }
                completion:^(BOOL finished){
                    if (finished) {
                        NSLog(@"Done!");
                    }
                }
];
```

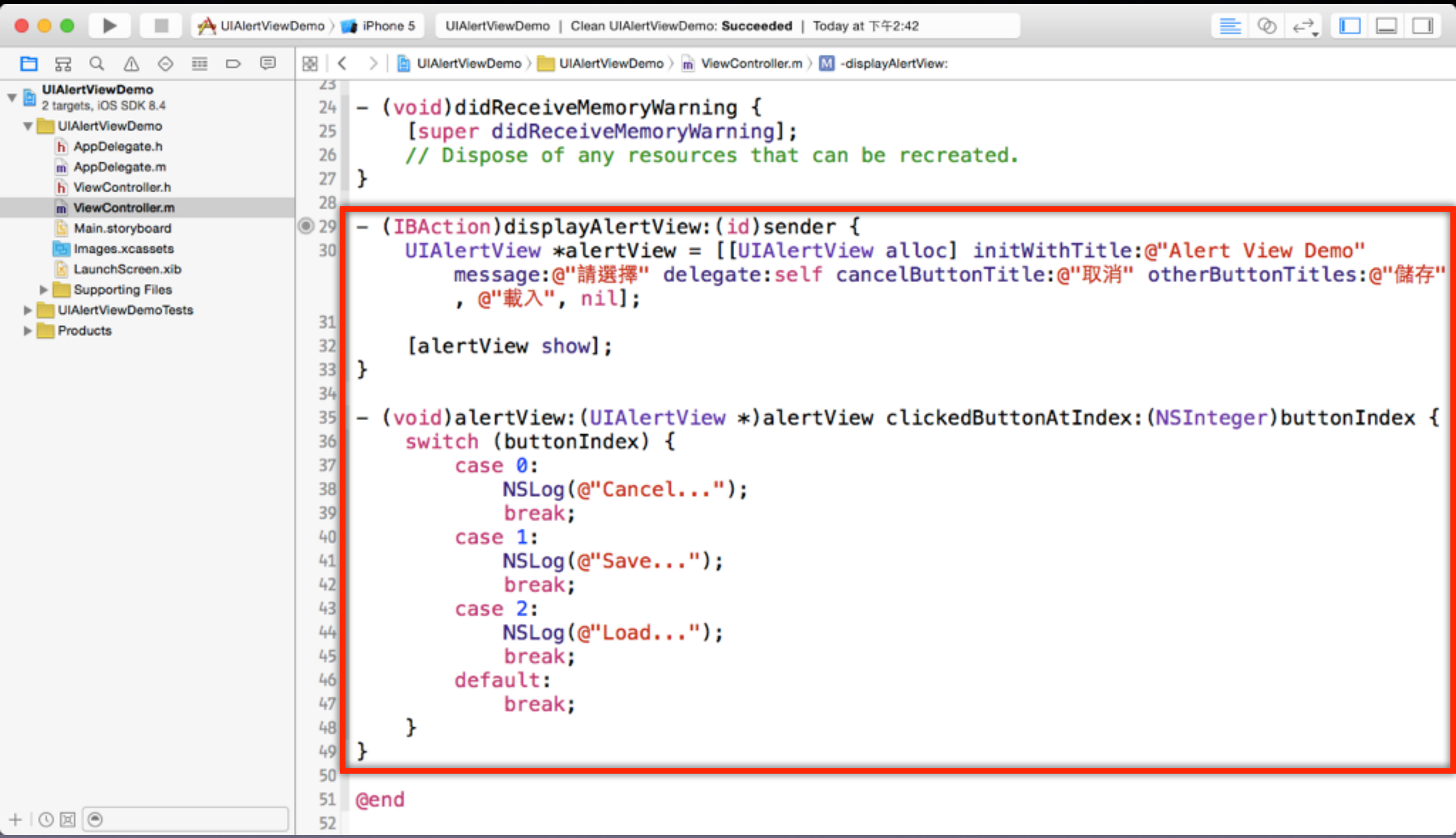
```objc
// 由一個 UIView 轉到另外一個 UIView
[UIView transitionFromView:view1
                    toView:view2
                  duration:0.5f
                   options:(UIViewAnimationOptionTransitionFlipFromLeft)
                animations:^{
                    self.view.alpha = 1;
                }
                completion:^(BOOL finished){
                    if (finished) {
                        NSLog(@"Done!");
                    }
                }
];
```

選擇與回應元件

- 對話盒元件 UIAlertView




```
//
//  ViewController.m
//  UIAlertViewDemo
//
//  Created by Brian Wang on 2015/8/24.
//  Copyright (c) 2015年 MapKing. All rights
    reserved.
//

#import "ViewController.h"

@interface ViewController ()

- (IBAction)displayAlertView:(id)sender;

@end

@implementation ViewController

- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the
        view, typically from a nib.
}

- (void)didReceiveMemoryWarning {
    [super didReceiveMemoryWarning];
    // Dispose of any resources that can be
        recreated.
}

- (IBAction)displayAlertView:(id)sender {
```


UIAlertViewDemo
iPhone 5
UIAlertViewDemo | Clean UIAlertViewDemo: Succeeded | Today at 下午2:42
UIAlertViewDemo
UIAlertViewDemo
ViewController.m
-displayAlertView:
UIAlertViewDemo
2 targets, iOS SDK 8.4
UIAlertViewDemo
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Images.xcassets
LaunchScreen.xib
Supporting Files
UIAlertViewDemoTests
Products
- (void)didReceiveMemoryWarning {
[super didReceiveMemoryWarning];
// Dispose of any resources that can be recreated.
}
- (IBAction)displayAlertView:(id)sender {
UIAlertView *alertView = [[UIAlertView alloc] initWithTitle:@"Alert View Demo"
message:@"請選擇" delegate:self cancelButtonTitle:@"取消" otherButtonTitles:@"儲存"
, @"載入", nil];
[alertView show];
}
- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex {
switch (buttonIndex) {
case 0:
NSLog(@"Cancel...");
break;
case 1:
NSLog(@"Save...");
break;
case 2:
NSLog(@"Load...");
break;
default:
break;
}
}
@end

LoginDemo 〉 iPhone 5 | Running LoginDemo on iPhone 5

LoginDemo 〉 LoginDemo 〉 ViewController.m 〉 -alertView:clickedButtonAtIndex:

- LoginDemo
  - 2 targets, iOS SDK 8.4
  - LoginDemo
    - AppDelegate.h
    - AppDelegate.m
    - ViewController.h
    - ViewController.m
    - Main.storyboard
    - Images.xcassets
    - LaunchScreen.xib
    - Supporting Files
  - LoginDemoTests
  - Products

```objc
- (IBAction)displayLoginAlertView:(id)sender {
    UIAlertView *loginAlertView = [[UIAlertView alloc] initWithTitle:@"登入系統" message:
        nil delegate:self cancelButtonTitle:@"取消" otherButtonTitles:@"登入", nil];
    loginAlertView.alertViewStyle = UIAlertViewStyleLoginAndPasswordInput;

    [loginAlertView show];
}

- (void)willPresentAlertView:(UIAlertView *)alertView {
    UITextField *usernameField = [alertView textFieldAtIndex:0];
    usernameField.placeholder = @"請輸入帳號";

    UITextField *passwordField = [alertView textFieldAtIndex:1];
    passwordField.placeholder = @"請輸入密碼";
}

- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex {
    UITextField *usernameField = [alertView textFieldAtIndex:0];
    UITextField *passwordField = [alertView textFieldAtIndex:1];

    switch (buttonIndex) {
        case 0:
            NSLog(@"Cancel...");
            break;
        case 1:
            NSLog(@"username=%@, password=%@", usernameField.text, passwordField.text);
            break;
        default:
            break;
    }
```

LoginDemo 〉 Thread 1 〉 9 main

- 動作清單元件 UIActionSheet

UIActio...etDemo 〉 iPhone 6 | UIActionSheetDemo: **Ready** | Today at 下午4:09

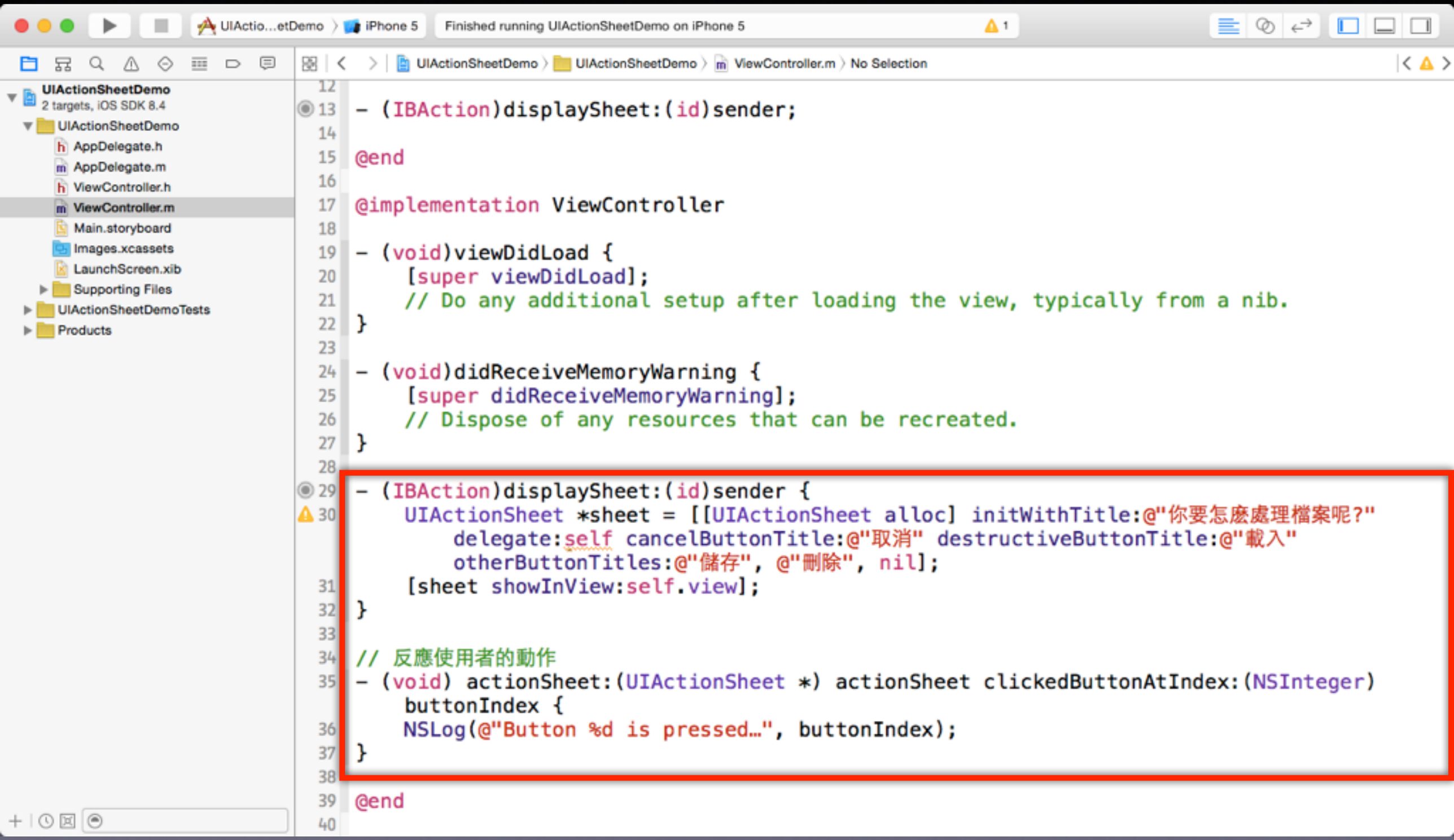
UIActionSheetDemo
2 targets, iOS SDK 8.4
- UIActionSheetDemo
  - AppDelegate.h
  - AppDelegate.m
  - ViewController.h
  - ViewController.m
  - Main.storyboard
  - Images.xcassets
  - LaunchScreen.xib
  - Supporting Files
- UIActionSheetDemoTests
- Products

UIActionSheetDemo 〉 Main.storyboard (Base) 〉 No Selection

wAny hAny

A... 〉 ViewController.m 〉 @interface ViewController() 〉 2

```
1  //
2  //  ViewController.m
3  //  UIActionSheetDemo
4  //
5  //  Created by Brian Wang on 2015/8/26.
6  //  Copyright (c) 2015年 MapKing. All
       rights reserved.
7  //
8
9  #import "ViewController.h"
10
11 @interface ViewController ()
12
13 - (IBAction)displaySheet:(id)sender;
14
15 @end
16
17 @implementation ViewController
18
19 - (void)viewDidLoad {
20     [super viewDidLoad];
21     // Do any additional setup after
           loading the view, typically from
           a nib.
22 }
23
24 - (void)didReceiveMemoryWarning {
25     [super didReceiveMemoryWarning];
26     // Dispose of any resources that can
           be recreated.
27 }
28
```


UIActio...etDemo
iPhone 5
Finished running UIActionSheetDemo on iPhone 5
UIActionSheetDemo
UIActionSheetDemo
ViewController.m
No Selection
UIActionSheetDemo
2 targets, iOS SDK 8.4
UIActionSheetDemo
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Images.xcassets
LaunchScreen.xib
Supporting Files
UIActionSheetDemoTests
Products
- (IBAction)displaySheet:(id)sender;
@end
@implementation ViewController
- (void)viewDidLoad {
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
}
- (void)didReceiveMemoryWarning {
[super didReceiveMemoryWarning];
// Dispose of any resources that can be recreated.
}
- (IBAction)displaySheet:(id)sender {
UIActionSheet *sheet = [[UIActionSheet alloc] initWithTitle:@"你要怎麼處理檔案呢?"
delegate:self cancelButtonTitle:@"取消" destructiveButtonTitle:@"載入"
otherButtonTitles:@"儲存", @"刪除", nil];
[sheet showInView:self.view];
}
// 反應使用者的動作
- (void) actionSheet:(UIActionSheet *) actionSheet clickedButtonAtIndex:(NSInteger)
buttonIndex {
NSLog(@"Button %d is pressed…", buttonIndex);
}
@end

- 日期選擇元件 UIDatePicker
  - UIDatePickerModeTime：時間選擇模式
  - UIDatePickerModeDate：日期選擇模式
  - UIDatePickerModeDateAndTime：日期加上時間選擇模式
  - UIDatePickerModeCountDownTimer：倒數計時模式

UIDatePickerDemo 〉 iPhone 6 | UIDatePickerDemo: **Ready** | Today at 下午4:29

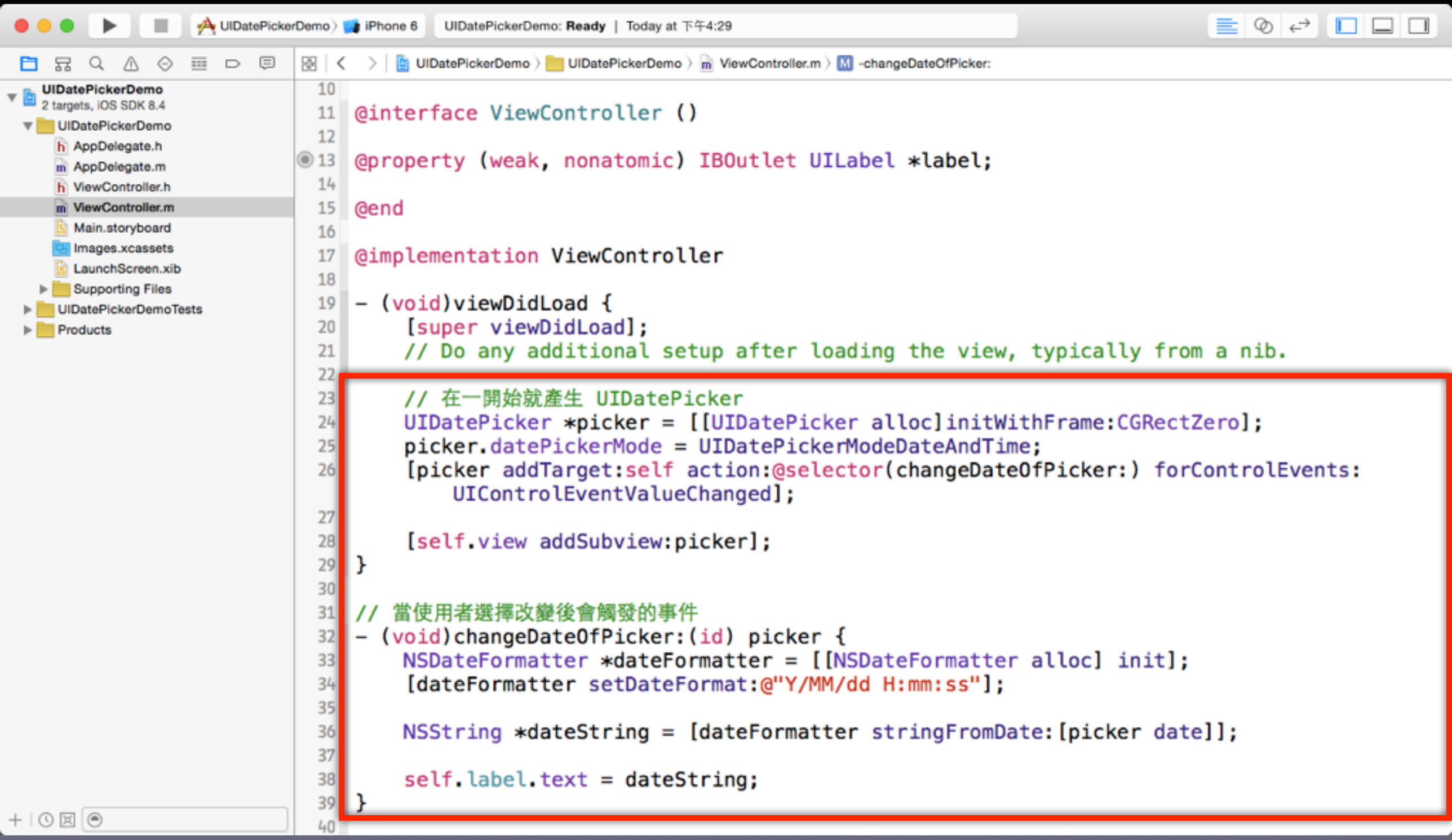
UIDatePickerDemo
2 targets, iOS SDK 8.4
- UIDatePickerDemo
  - AppDelegate.h
  - AppDelegate.m
  - ViewController.h
  - ViewController.m
  - Main.storyboard
  - Images.xcassets
  - LaunchScreen.xib
  - Supporting Files
- UIDatePickerDemoTests
- Products

UIDatePickerDemo 〉 View 〉 Label

wAny hAny

A.. 〉 ViewController.m 〉 @interface ViewController() 〉 2

```
1  //
2  //  ViewController.m
3  //  UIDatePickerDemo
4  //
5  //  Created by Brian Wang on 2015/8/26.
6  //  Copyright (c) 2015年 MapKing. All
       rights reserved.
7  //
8
9  #import "ViewController.h"
10
11 @interface ViewController ()
12
13 @property (weak, nonatomic) IBOutlet
       UILabel *label;
14
15 @end
16
17 @implementation ViewController
18
19 - (void)viewDidLoad {
20     [super viewDidLoad];
21     // Do any additional setup after
           loading the view, typically from
           a nib.
22 }
23
24 - (void)didReceiveMemoryWarning {
25     [super didReceiveMemoryWarning];
26     // Dispose of any resources that can
           be recreated.
27 }
```


UIDatePickerDemo
iPhone 6
UIDatePickerDemo: Ready | Today at 下午4:29
UIDatePickerDemo
UIDatePickerDemo
ViewController.m
-changeDateOfPicker:
UIDatePickerDemo
2 targets, iOS SDK 8.4
UIDatePickerDemo
AppDelegate.h
AppDelegate.m
ViewController.h
ViewController.m
Main.storyboard
Images.xcassets
LaunchScreen.xib
Supporting Files
UIDatePickerDemoTests
Products
@interface ViewController ()
@property (weak, nonatomic) IBOutlet UILabel *label;
@end
@implementation ViewController
- (void)viewDidLoad {
[super viewDidLoad];
// Do any additional setup after loading the view, typically from a nib.
// 在一開始就產生 UIDatePicker
UIDatePicker *picker = [[UIDatePicker alloc]initWithFrame:CGRectZero];
picker.datePickerMode = UIDatePickerModeDateAndTime;
[picker addTarget:self action:@selector(changeDateOfPicker:) forControlEvents:
UIControlEventValueChanged];
[self.view addSubview:picker];
}
// 當使用者選擇改變後會觸發的事件
- (void)changeDateOfPicker:(id) picker {
NSDateFormatter *dateFormatter = [[NSDateFormatter alloc] init];
[dateFormatter setDateFormat:@"Y/MM/dd H:mm:ss"];
NSString *dateString = [dateFormatter stringFromDate:[picker date]];
self.label.text = dateString;
}

- 滾輪元件 UIPickerView
  - Components - 欄位
  - Rows - 欄位內資料數量
  - 當前所選擇為哪一項

    - (void)pickerView:(UIPickerView *)pickerView didSelectRow:(NSInteger)row inComponent:(NSInteger)component;

- 設定滾輪總共有幾個欄位

  - (NSInteger)numberOfComponentsInPickerView:(UIPickerView *)pickerView;

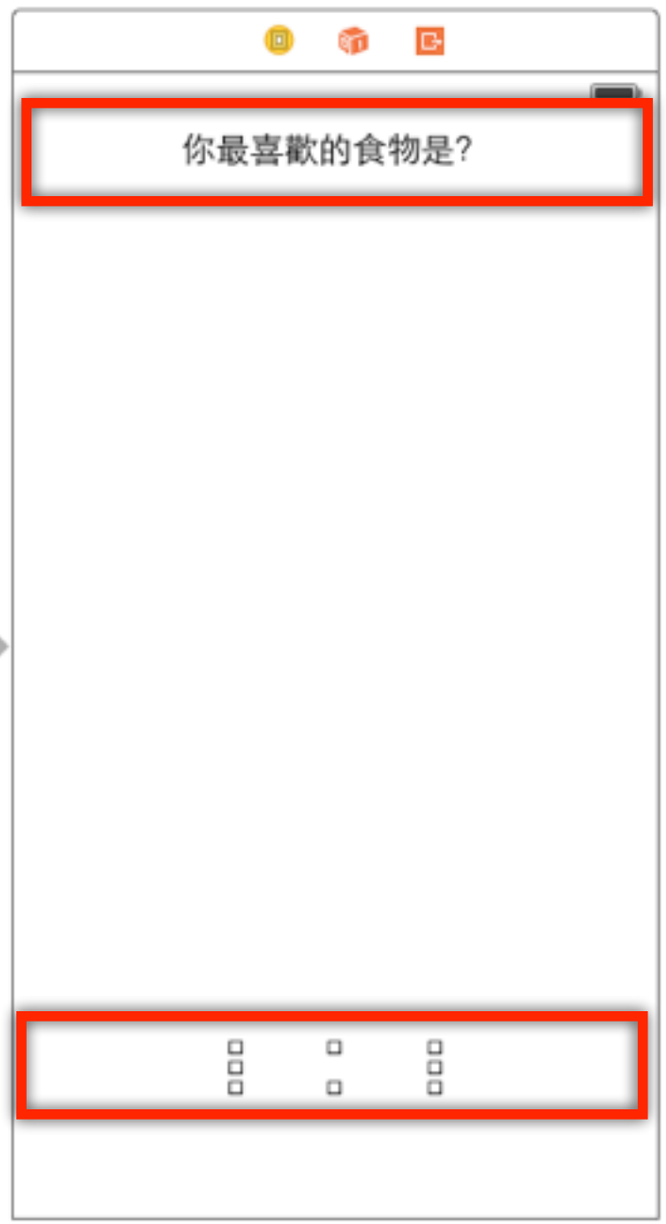
- 設定某個欄位有幾筆資料
  - (NSInteger)pickerView:(UIPickerView *)pickerView numberOfRowsInComponent:(NSInteger)component;
- 設定滾輪顯示的文字
  - (NSString *)pickerView:(UIPickerView *)pickerView titleForRow:(NSInteger)row forComponent:(NSInteger)component;

UIPickerViewDemo 〉 iPhone 6 | UIPickerViewDemo: **Ready** | Today at 下午4:49

UIPickerViewDemo 〉 View 〉 Label

A. 〉 ViewController.m 〉 @interface ViewController() 〉 2

- UIPickerViewDemo
  - 2 targets, iOS SDK 8.4
  - UIPickerViewDemo
    - AppDelegate.h
    - AppDelegate.m
    - ViewController.h
    - ViewController.m
    - Main.storyboard
    - Images.xcassets
    - LaunchScreen.xib
    - Supporting Files
  - UIPickerViewDemoTests
  - Products



wAny hAny

```
1  //
2  //  ViewController.m
3  //  UIPickerViewDemo
4  //
5  //  Created by Brian Wang on 2015/8/26.
6  //  Copyright (c) 2015年 MapKing. All
       rights reserved.
7  //
8
9  #import "ViewController.h"
10
11 @interface ViewController ()
12
13 @property (weak, nonatomic) IBOutlet
       UILabel *label;
14
15 @end
16
17 @implementation ViewController
18
19 - (void)viewDidLoad {
20     [super viewDidLoad];
21     // Do any additional setup after
           loading the view, typically from
           a nib.
22 }
23
24 - (void)didReceiveMemoryWarning {
25     [super didReceiveMemoryWarning];
26     // Dispose of any resources that can
           be recreated.
27 }
```

```objc
#import <UIKit/UIKit.h>

@interface ViewController : UIViewController {
    NSMutableDictionary *data;
    NSArray *keys;
}


@end
```

```objc
// 當使用者選中某個 component 內的某筆資料時
- (void)pickerView:(UIPickerView *)thePickerView didSelectRow:(NSInteger)row
    inComponent:(NSInteger)component {
    if(component == 0) {
        // 重新載入 component = 1 的畫面
        [thePickerView reloadComponent:1];
    } else {
        NSString *key = [keys objectAtIndex:[thePickerView selectedRowInComponent:
            0]]; // 飲料或甜點
        NSArray *array = [data objectForKey:key];
        self.label.text = [array objectAtIndex:[thePickerView selectedRowInComponent:1]]
            ;
    }
}

// 設定滾輪總共有幾個欄位
- (NSInteger)numberOfComponentsInPickerView:(UIPickerView *)thePickerView {
    return 2;
}

//設定滾輪總共有幾個項目
- (NSInteger)pickerView:(UIPickerView *)thePickerView numberOfRowsInComponent:(NSInteger
    )component {
    if(component == 0) {
        return [keys count];
    } else {
        NSString *key = [keys objectAtIndex:[thePickerView selectedRowInComponent:
            0]]; // 飲料或甜點
        NSArray *array = [data objectForKey:key];
        return [array count];
    }
}
```

```objc
// 設定滾輪顯示的文字
- (NSString *)pickerView:(UIPickerView *)thePickerView titleForRow:(NSInteger)row
    forComponent:(NSInteger)component {
    if(component == 0) {
        return[keys objectAtIndex:row];
    } else {
        NSString *key = [keys objectAtIndex:[thePickerView selectedRowInComponent:
            0]]; // 飲料或甜點
        NSArray *array = [data objectForKey:key];
        return [array objectAtIndex:row];
    }
}

- (void)viewDidUnload {
    [super viewDidUnload];
    self.label = nil;
}
```

```objc
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.

    [self prepareData];

    UIPickerView *picker = [[UIPickerView alloc]initWithFrame:CGRectZero];
    picker.transform = CGAffineTransformMakeTranslation(0, 50);
    picker.delegate = self;
    picker.showsSelectionIndicator = YES;
    [picker selectRow:0 inComponent:0 animated:YES];
    [picker selectRow:0 inComponent:1 animated:YES];

    [self.view addSubview:picker];
}
```

```objc
- (void)prepareData {
    data = [[NSMutableDictionary alloc] init];
    [data setValue:[NSArray arrayWithObjects:@"可樂", @"沙士", @"果汁", @"其他", nil]
        forKey:@"飲料"];
    [data setValue:[NSArray arrayWithObjects:@"蛋糕", @"糖果", @"巧克力", @"其他", nil]
        forKey:@"甜點"];
    keys = [[data allKeys]
            sortedArrayUsingComparator:(NSComparator)^(id obj1, id obj2) {
                return [obj1 caseInsensitiveCompare:obj2];
            }];
}
```

狀態顯示元件

- 告知使用者目前應用程式的狀態
- 活動狀態指示元件 UIActivityIndicatorView
  - 無法預估結束時間

```objc
// 產生動作指示動畫
- (void)createIndicator {
    indicator = [[UIActivityIndicatorView alloc] initWithFrame:CGRectZero];
    indicator.center = CGPointMake(160.0f,208.0f);
    indicator.hidesWhenStopped = YES;
    indicator.color = [UIColor redColor];
    indicator.activityIndicatorViewStyle = UIActivityIndicatorViewStyleGray;
    [self.view addSubview:indicator];
}

// 顯示或隱藏進度動畫
- (IBAction)showIndicator:(id)sender {
    if(indicator) {
        [self createIndicator];
    }

    if([indicator isAnimating]) {
        [indicator stopAnimating];
    } else {
        [indicator startAnimating];
    }
}
```

- 進度狀態指示元件 UIProgressView
  - 可以以預估執行時間的進度

```
- (void)changeValue:(id)sender {
    if (value++ >= 10) {
        [sender invalidate];
    } else {
        UIProgressView *progress = (UIProgressView *) [sender userInfo];
        progress.progress = value / 10.0f;
    }
}

- (IBAction)startProgress:(id)sender {
    UIProgressView *progress = [[UIProgressView alloc] initWithFrame:CGRectMake(0,
        450.0f, 320.0f, 30.0f)];

    [self.view addSubview:progress];

    progress.progressViewStyle = UIProgressViewStyleDefault;
    value = 0.0f;
    [NSTimer scheduledTimerWithTimeInterval:0.2f target:self selector:@selector
        (changeValue:)userInfo:progress repeats:YES];
}
```

- 標記 (Badge) 的使用
  - 標記通常會與推播的技術一起使用
    - [UIApplication sharedApplication].applicationIconBadgeNumber = 1;

文字輸入與顯示元件

- 文字顯示元件 UILabel
  - 顯示文字，但不可輸入文字
  - 斷行方式
    - 自動斷行，需配合 numberOfLines 屬性設定行數
      label.lineBreakMode = UILineBreakModeWordWrap;

- 自動在結尾加上 ”...” 符號
  label.lineBreakMode = UILineBreakModeTailTruncation;

- 文字大小自動調整
  label.numberOfLines = 1;
  label.adjustFontSizeToFitWidth = YES;

- 單行文字輸入元件 UITextField
  - 輸入焦點移動到文字輸入框時，虛擬鍵盤會自動跳出
  - UITextField 變成 FirstResponder (呼叫 becomeFirstResponder 方法) 時虛擬鍵盤出現
  - 呼叫 resignFirstResponder 方法就會消失

- 多行文字輸入元件 UITextView
  - 顯示長篇文章
  - 可指定顏色與字型
  - 可用來編輯文字
  - 會自動產生捲軸
  - 有個相對應的 UITextViewDelegate

- - (BOOL)textViewShouldBeginEditing:(UITextView *)textView;
- 當使用者準備開始編輯文字時會觸發這個事件
- 依據這個事件的回傳值決定是否可以允許編輯

- - (BOOL)textViewShouldEndEditing:(UITextView *)textView;
  - 當使用者結束編輯將焦點轉移到另一個控制元件時會觸發這個事件
  - 依據這個事件的回傳值決定是否可以將焦點轉移到別的元件上面
  - 可以在這個事件內先檢查使用者的輸入是否為無效值
  - 若是無效值則可以傳回 NO 讓使用者重新輸入有效值

- - (void)textViewDidBeginEditing:(UITextView *)textView;
  - 當使用者準備開始編輯，並且尚未輸入任何文字時會觸發這個事件
- - (void)textViewDidEndEditing:(UITextView *)textView;
  - 當使用者結束編輯，且焦點轉移到另一個控制元件後所觸發的事件

- - (BOOL)textView:(UITextView *)textView shouldChangeTextInRange:(NSRange)range replacementText:(NSString *)text;
  - 回傳是否可以取代某個範圍內的文字
- - (void)textViewDidChange:(UITextView *)textView;
  - 當這個元件裡面的文字被改變後所觸發的事件

- - (void)textViewDidChangeSelection:(UITextView *)textView;
- 當使用者選擇的文字被改變後所觸發的事件

其他控制元件

- 一般按鈕 UIButton
  - 一些常見的按鈕類型被定義在 UIButtonType 這個列舉型態內
- 允許使用外部的圖形來作為按鈕的圖形或背景
- iOS 允許按鈕在不同狀態下可以使用不同的圖形來表示

- 一般狀態 (UIControlStateNormal)
- 被按下的狀態 (UIControlStateHighlighted)
- 無法使用的狀態 (UIControlStateDisabled)
- 使用 setImage: forState: 來設定不同狀態下的圖形

- 開關元件 UISwitch
  - 狀態只有 Yes 與 NO 兩種狀態
  - 當切到 ON 的時候開關顏色會顯示為橘色或藍色
  - 反之切到 OFF 的時候會變成灰色狀態
  - 透過 on 這個點運算子來存取這個元件的狀態
  - 透過 setOn: animated: 讓開關在切換時加上動畫效果

- 滑軌元件 UISlider
  - 用於讓使用者設定某個設定值
  - 此設定值會被限制在某一個區段之間
  - 使用時必須先設定最小值 (minimumValue) 與最大值 (maximumValue)
  - 可以透過 continuous 屬性來設定元件的數值變化是否為連續性
  - 滑軌元件上面的按鈕稱為 thumbImage，可透過 setThumbImage: forState: 設定這個圖案

- 分頁元件 UIPageControl
  - 僅具有視覺上的意義
  - 一些按鈕的集合
  - 透過這個元件讓螢幕彷彿多了很多頁面
  - 元件不會完成畫面切換動作，必須自己來完成

- 分類元件 UISegmentedControl
  - 這元件的上面可以顯示多個按鈕
  - 但同時間只會有一個按鈕可以被按下
  - 透過 initWithItems: 建構一個分類元件
  - 在事件的處理上與 UIButton 相同，夠過 Target-Action 機制來實現

- 步進器元件 UIStepper
  - iOS 5 之後才提供的元件
  - 像是開關元件，但可以用來增減數值
  - autorepeat：(YES) 一直按住同一個按鍵會使得數值不斷改變，(NO) 每改變一次數值都得重新按一次按鍵

- continuous：(YES) 每次使用者按下 (或按住) 案件都會使數值立刻改變，(NO) 等到按鍵事件結束後才會改變數值

- wraps：(YES) 只要步進器的數值超過最大值 (maximumValue)，那其數值就會回到最小值 (minimumValue) 開始算起

- 改變數值可以透過 stepValue 來改變，而其數值則由 value 取得

認識 Responder