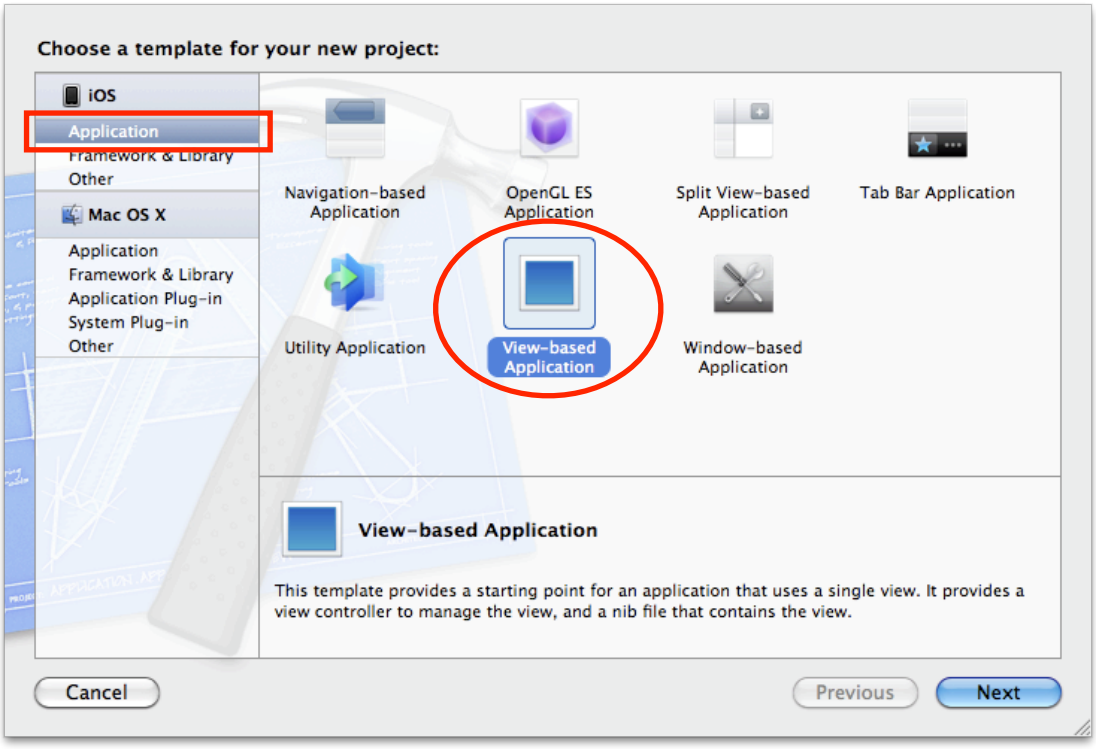
Lab FileReadWrite

Step 1. 在File>New>New Project開啓一個新的專案, 在iOS的Application目錄裡面選擇 view based application

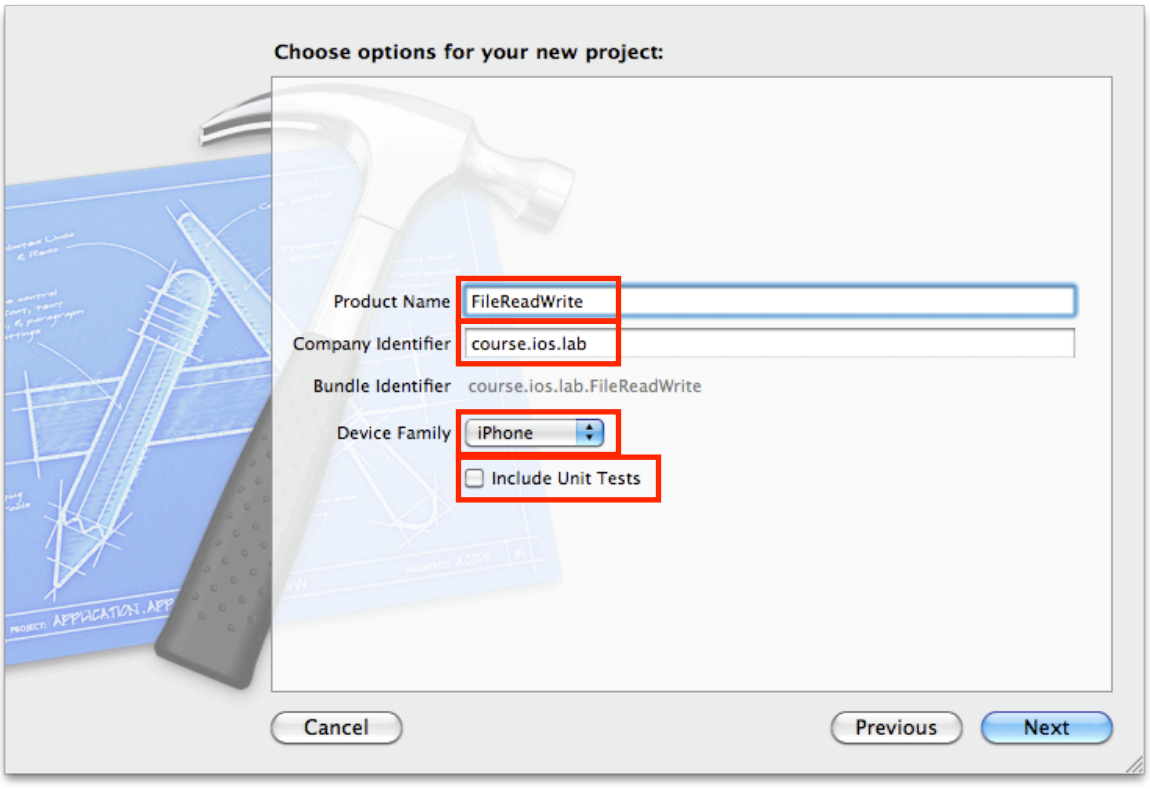


Step 2. 並將此專案命名為 **FileReadWrite**

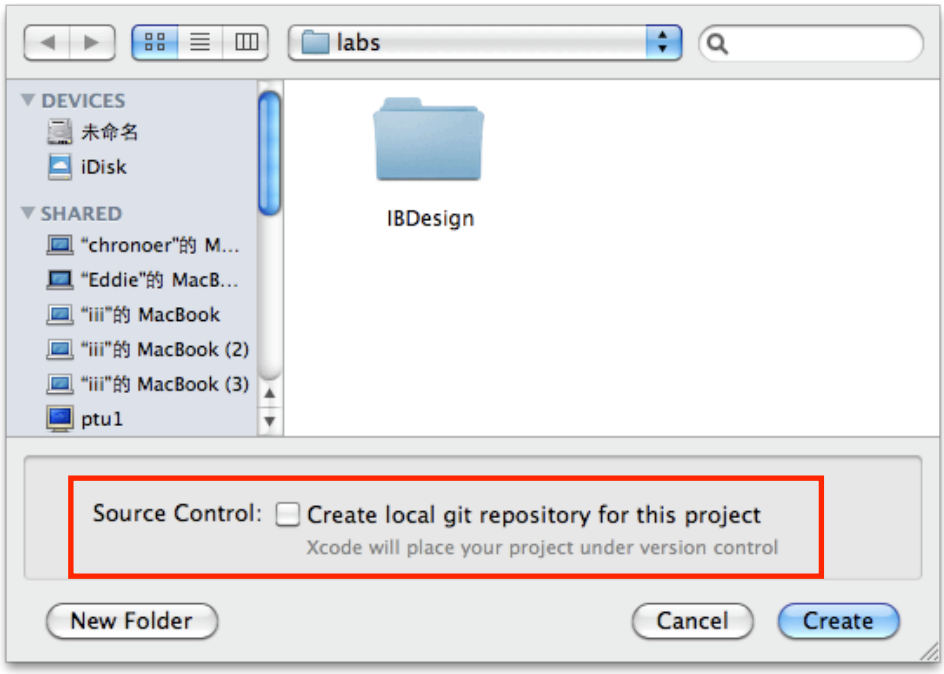
Company Identifier是填入Bundle的名稱,在此統一填入**course.ios.lab** (也可自行填入)

Device Family選擇**iPhone**

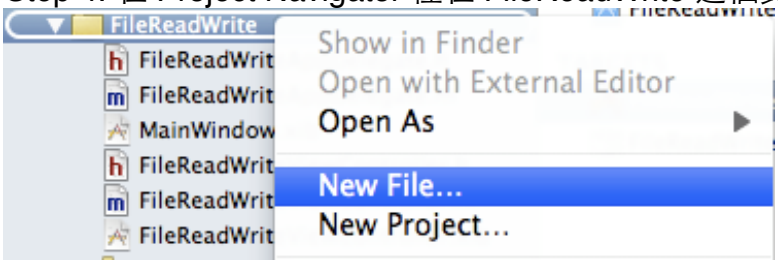
Include Unit Tests是做語意邏輯測試用,可勾選也可不勾選,在此我們統一**不勾選**



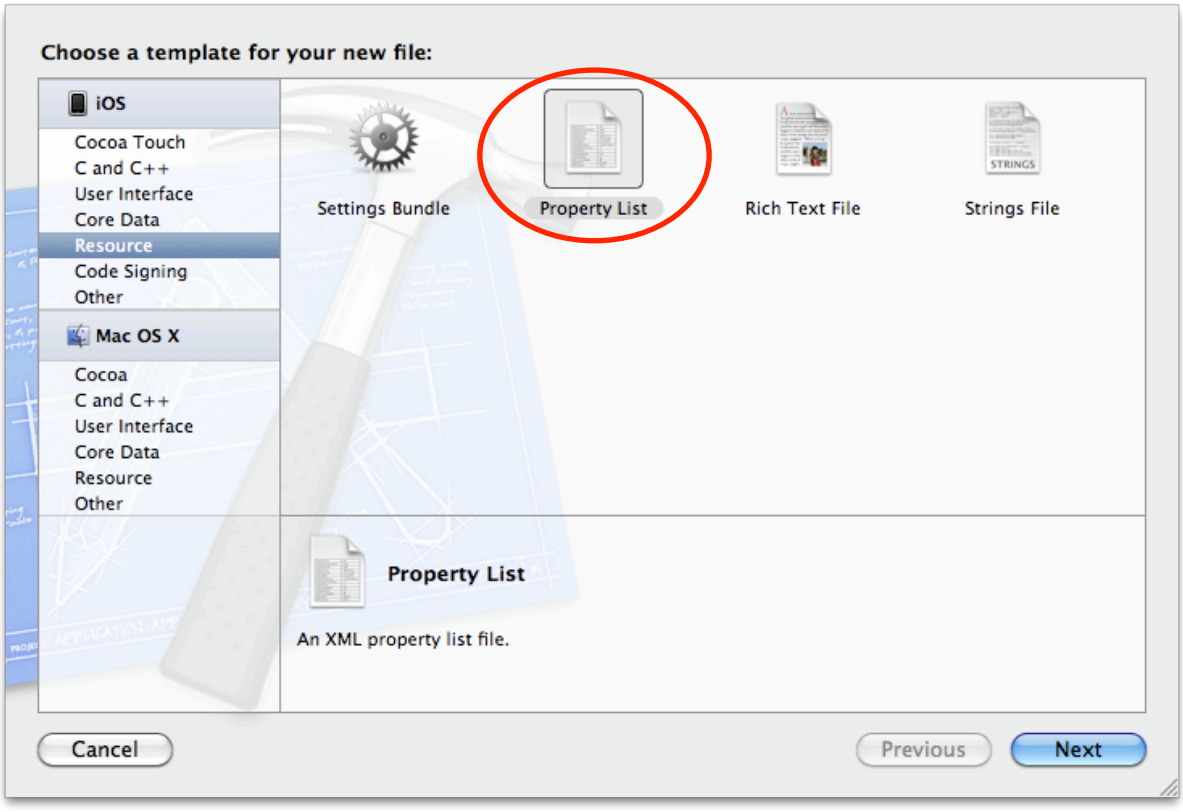
Step 3.選擇存檔的位置, 在此我們不做version control,統一不勾選Create local git repository for this project



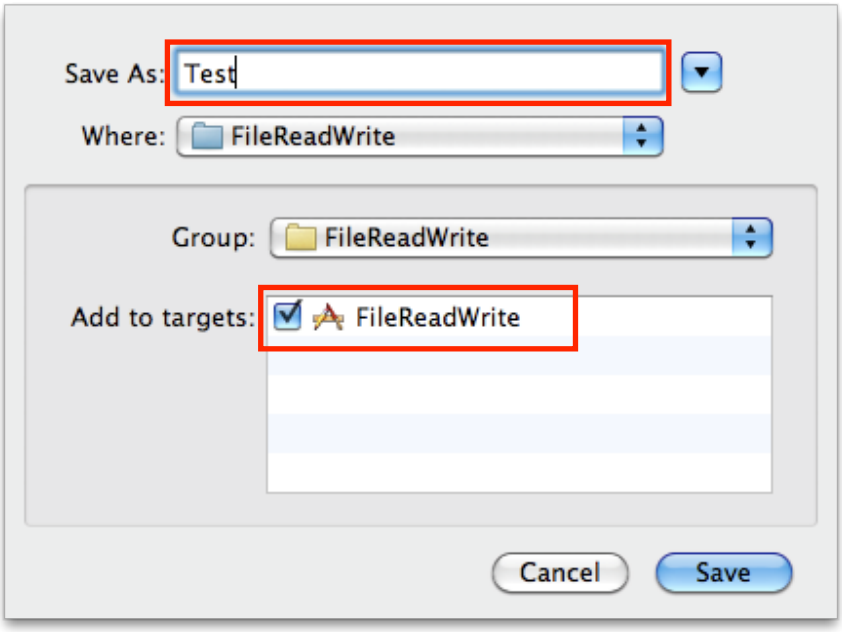
Step 4. 在 Project Navigator 裡在 FileReadWrite 這個資料夾上按右鍵來新增檔案



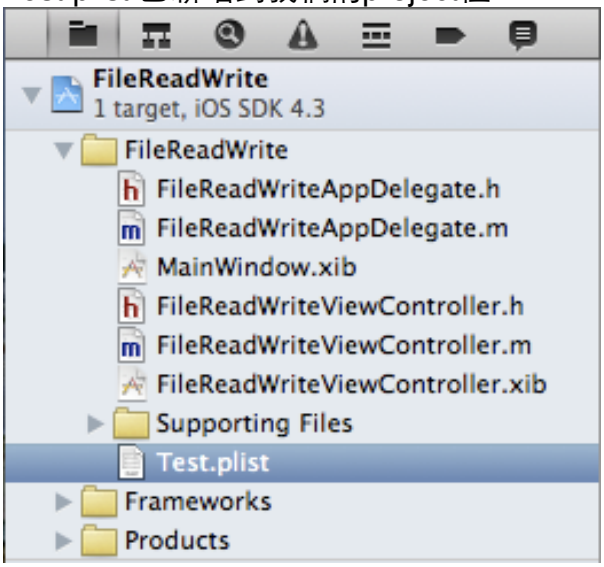
在iOS裡Resource裡面選擇Property List



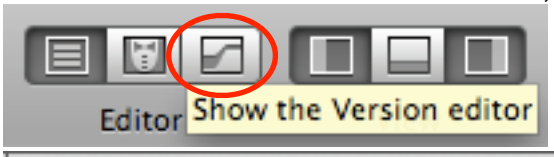
命名為 Test, 並Add to targets勾選 FileReadWrite

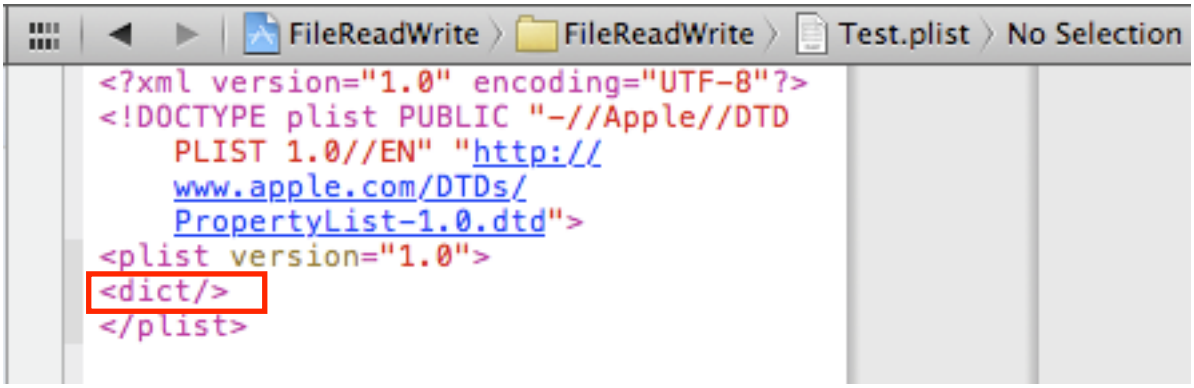


Test.plist 已新增到我們的project裡

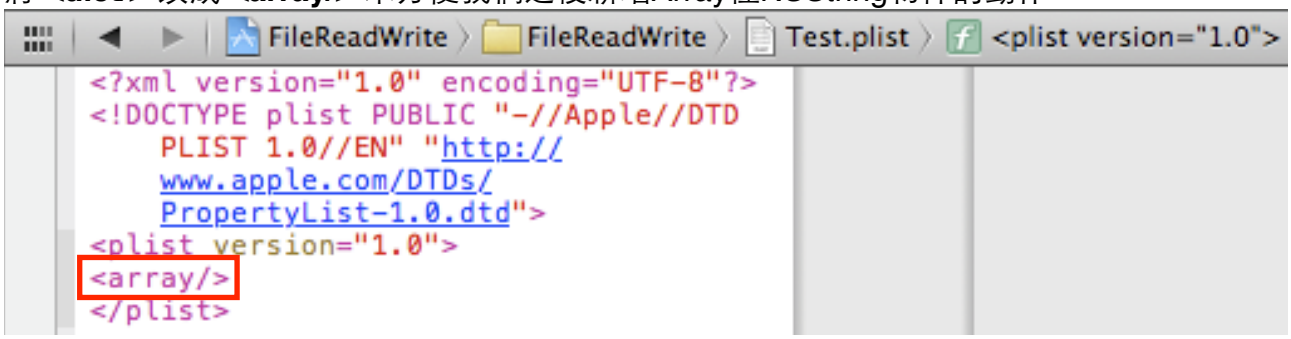


Step 5. 開啓剛新增 Test.plist
在右上角點選 Show the Version editor, 會看到這個plist的原始XML檔

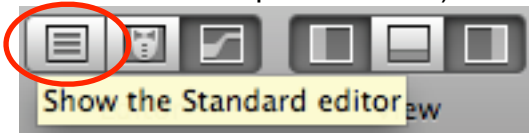




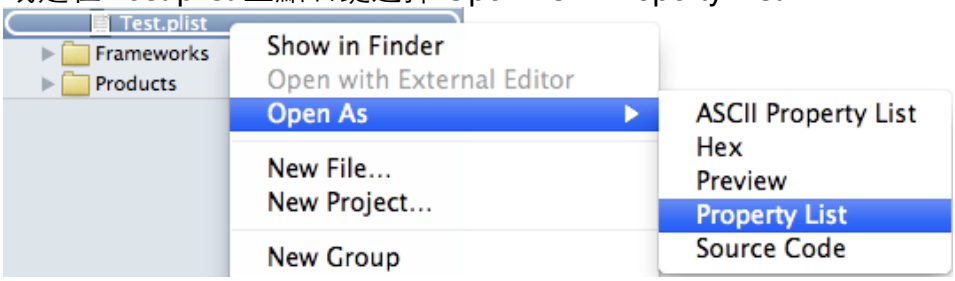
將 **<dict/>** 改成 **<array/>** 來方便我們之後新增Array裡NSString物件的動作



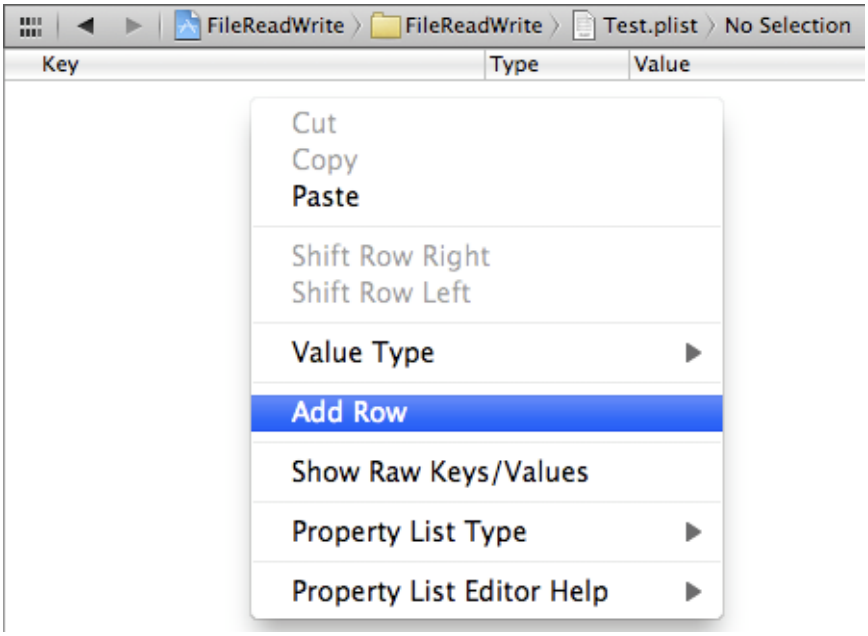
Step 6. 然後點回原來的 Show the Standard editor ,記得存檔(⌘+S)(注意可能要重新點回才回到一般Xcode的plist操作界面)



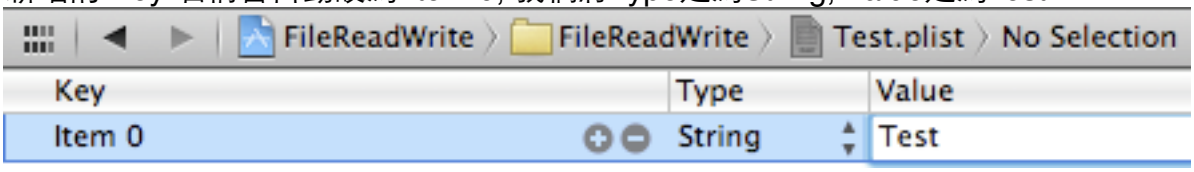
或是在 Test.plist 上點右鍵選擇 Open As > Property List



點右鍵選擇Add Row

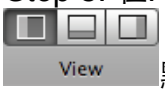


新增的 Key 名稱會自動設為 Item 0, 我們將Type定為String, Value定為Test

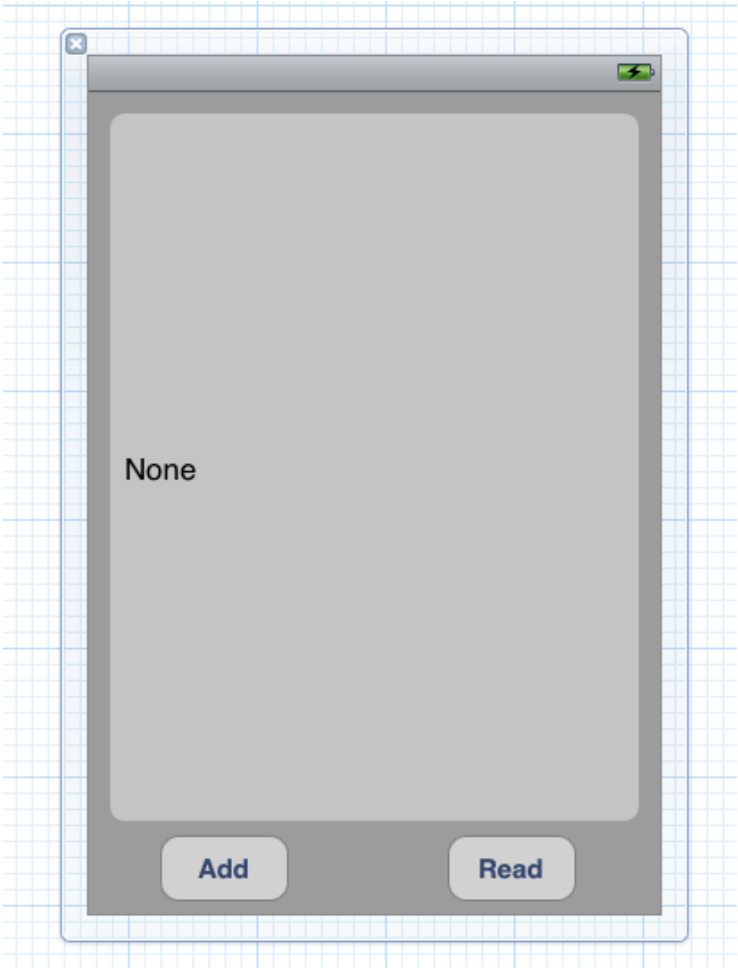


Step 7. 在FileReadWriteViewController.h裡加入View裡對應的物件, 物件對應的Action, 以及我們會用到的變數和method

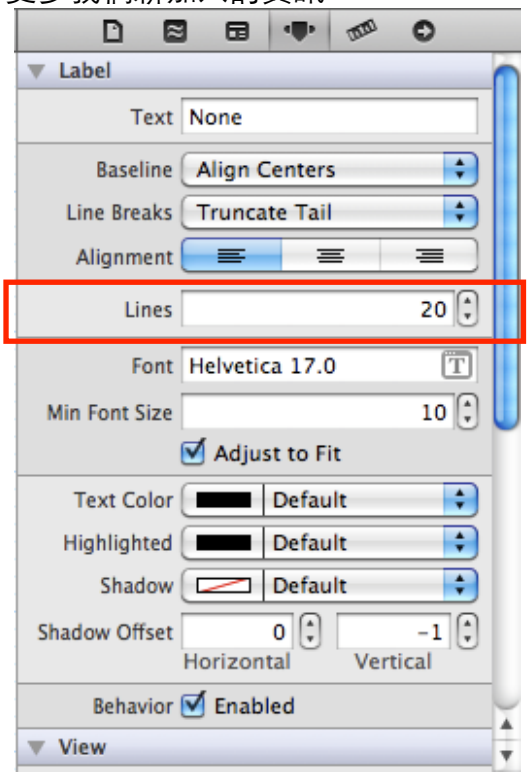
```
#import <UIKit/UIKit.h>

@interface FileReadWriteViewController : UIViewController {
     IBOutlet UILabel *myLabel;
     IBOutlet UIButton *myAddButton;
     IBOutlet UIButton *myReadButton;
     NSMutableString *information;
     NSMutableArray *setArray;
     NSMutableArray *newArray;
}
- (IBAction) add;
- (IBAction) read;
- (NSString *) fileLocation;
@end
```

Step 8. 在Project Navigator裡開啓FileReadWriteViewController.xib, 開啓在點選右上角的 

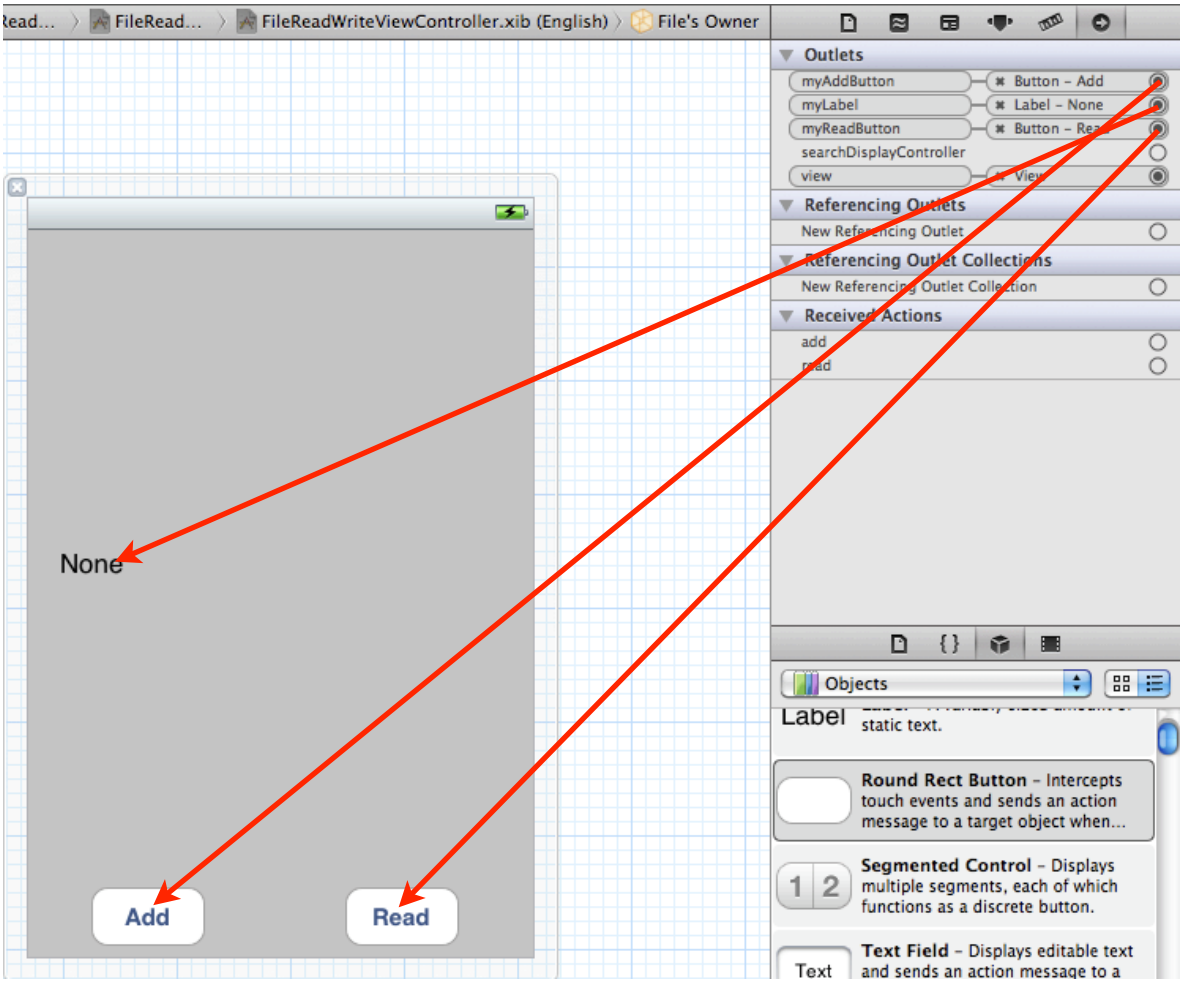
點下開啓右邊視窗分頁的  來開啓Inspectors 和 Libraries 加入我們需要的物件: 一個Label和兩個Button, 並將Label 預設的 text 設為None, 左邊的Button 的 Title 設為 Add, 右邊的Button 的 Title 設為 Read. 並將加入的Label的Frame盡量拉大, 以便顯示更多資訊,



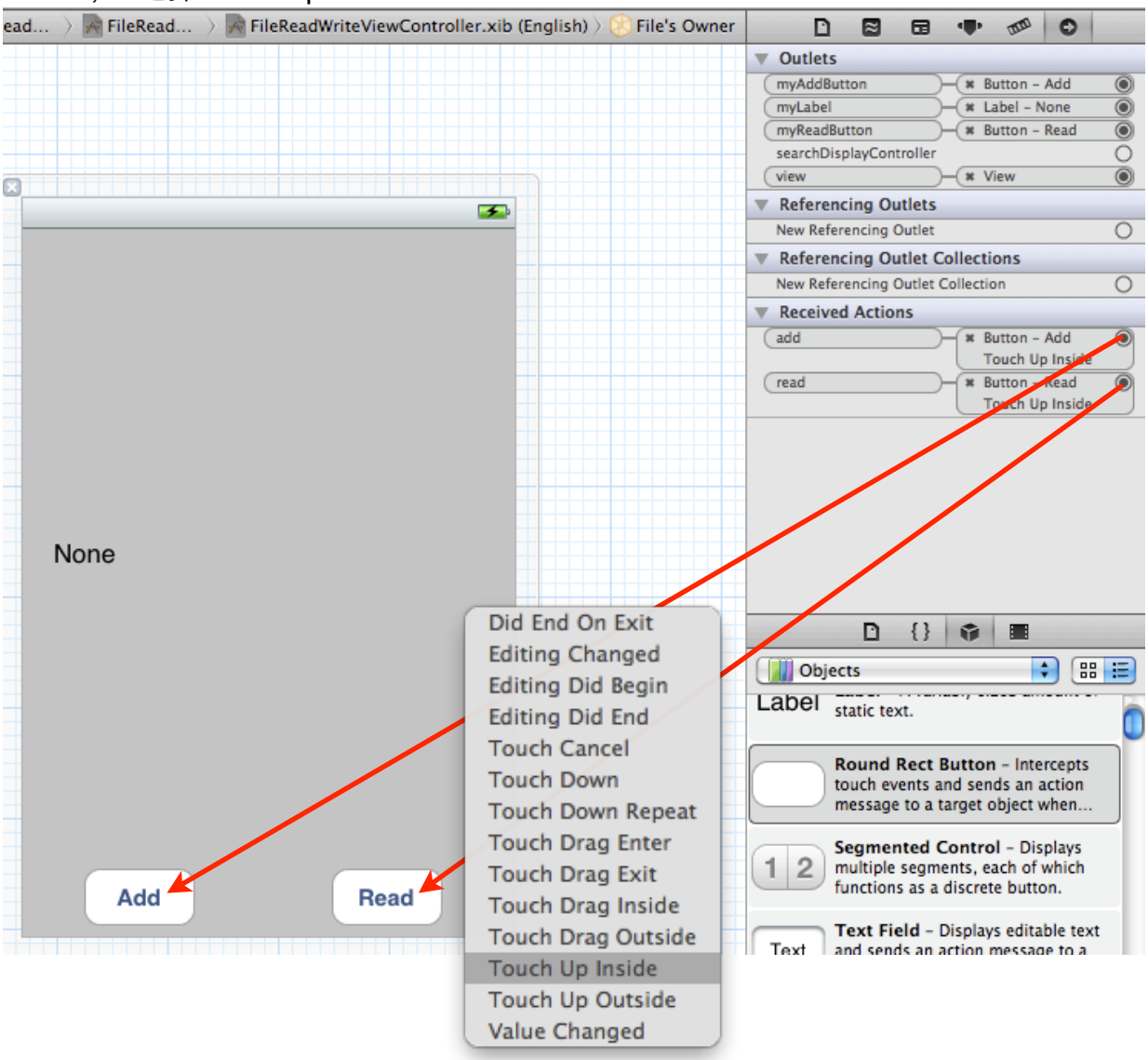
Step 9. 點選新增的Label, 在他的Attribute Inspector裡將Layout的Lines加到20行, 以便顯示更多我們新加入的資訊



Step 10. 點選File's Owner來開啓Connections Inspector來連結所有的宣告的物件 myLabel連到上放的Label; myAddButton連到左邊的**Add** Button; myReadButton連到右邊的 **Read** Button



Step 11. 連結對應的IBAction
add這個IBAction連到左邊的Button, 並選擇Touch Up Inside; read這個IBAction連到右邊的Button, 並選擇Touch Up Inside



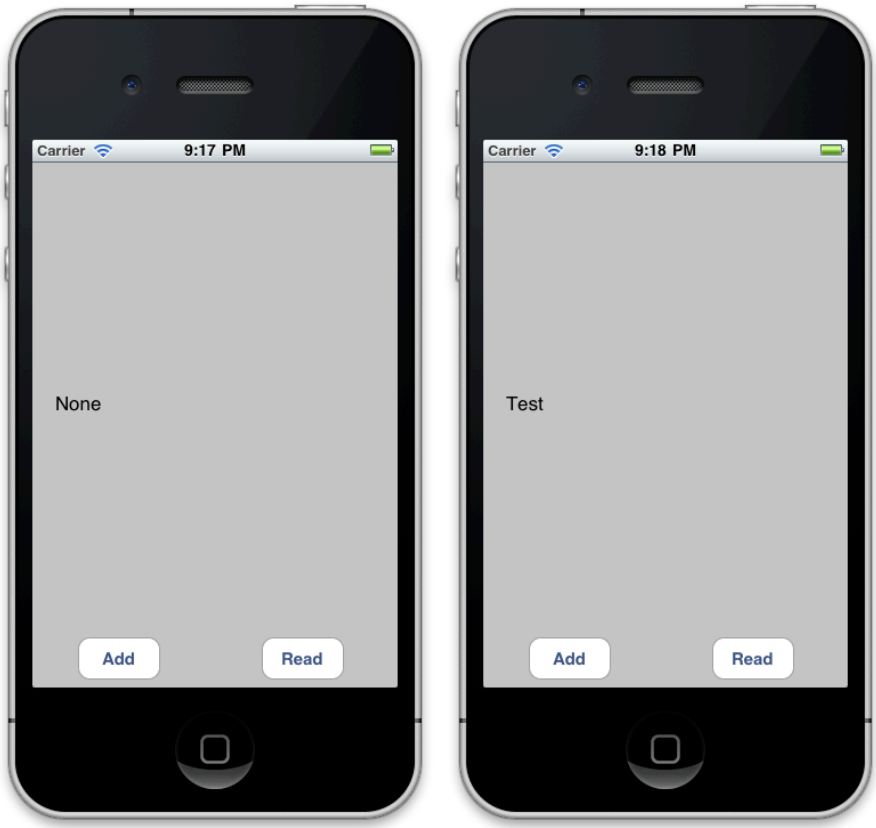
Step 12. 在 FileReadWriteViewController.m 裡, 將viewDidLoad這個Method的Mark去掉, 加入初始化顯示用的NSMutableString “information” 以及從剛剛新增的plist檔案的Array來作初始化的 NSMutableArray “setArray”

```
// Implement viewDidLoad to do additional setup after loading the view,
typically from a nib.
- (void)viewDidLoad {
     [super viewDidLoad];
     information = [[NSMutableString alloc] init];
     setArray = [[NSMutableArray alloc] initWithContentsOfFile:[self
fileLocation]];

}
```

Step 13. 同樣在 FileReadWriteViewController.m 裡, 我們實作對應兩個UIButton的IBAction, add這個method在剛剛初始化的setArray新增一個由現在時間產生的NSString物件 [[NSDate date] description]來顯示新增物件的時間, 並再寫入Test.plist這個檔案裡 read這個method則負責將Test.plist裡Array裡的每個String取出來並且顯示在myLabel裡 並實作fileLocation這個method去每次動態取得Test.plist位置的Path來給每次寫入和讀出使用

```
- (IBAction)add
{
     [setArray addObject:[@"Hello at " stringByAppendingString:[[NSDate
date] description]]];
     [setArray writeToFile:[self fileLocation] atomically:YES];
}

- (IBAction)read
{
     newArray = [[NSMutableArray alloc] initWithContentsOfFile:[self
fileLocation]];
     for(NSString *ele in newArray){
           [information appendString:[ele
stringByAppendingString:@"\n"]];
     }
     [myLabel setText:information];
     [information setString:@""];
}

- (NSString * ) fileLocation
{
     return [[NSBundle mainBundle] pathForResource:@"Test"
ofType:@"plist"];
}
```

Step 14. Run (⌘+R)
一開啓Label顯示預設text “None”,按一下Read, 顯示我們一開始寫入plist的第一個Item “Test”



按一下Add這個Button, 可加入現在時間的NSString物件並可再按一次Read這個Button來顯示出來,多按幾次Add Button, 再按Read Button可以顯示我們剛在不同時間加入的顯示時間的NSString Object