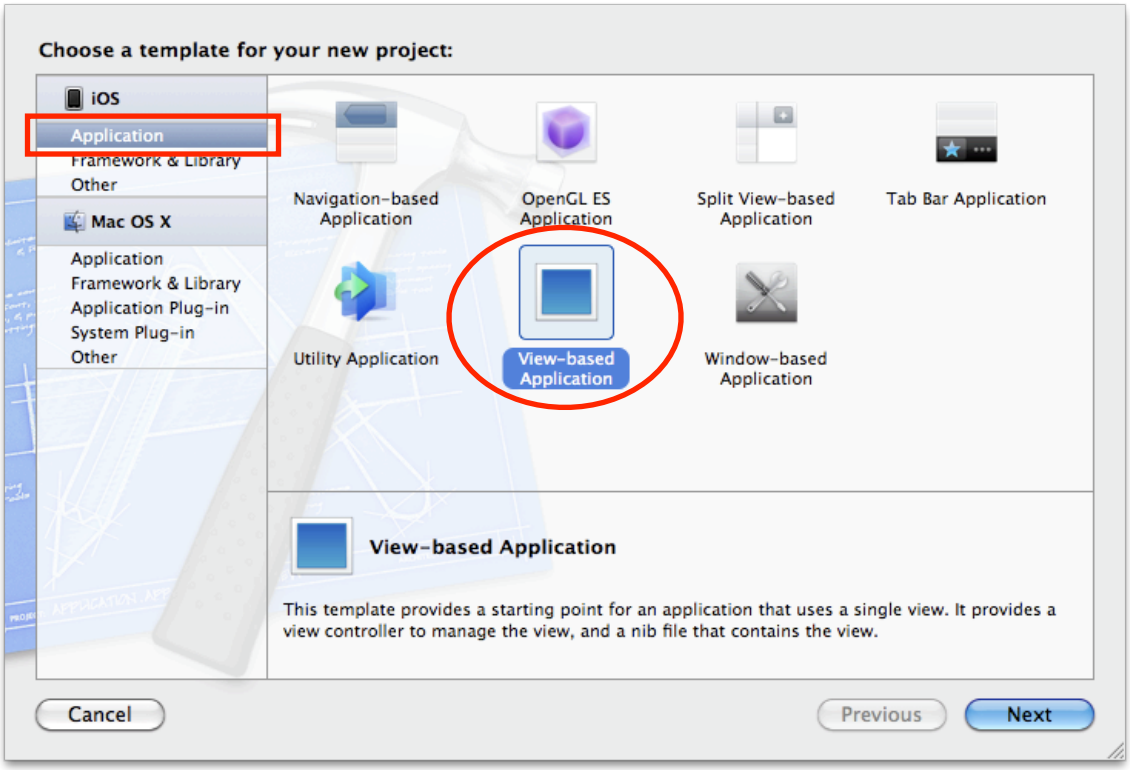
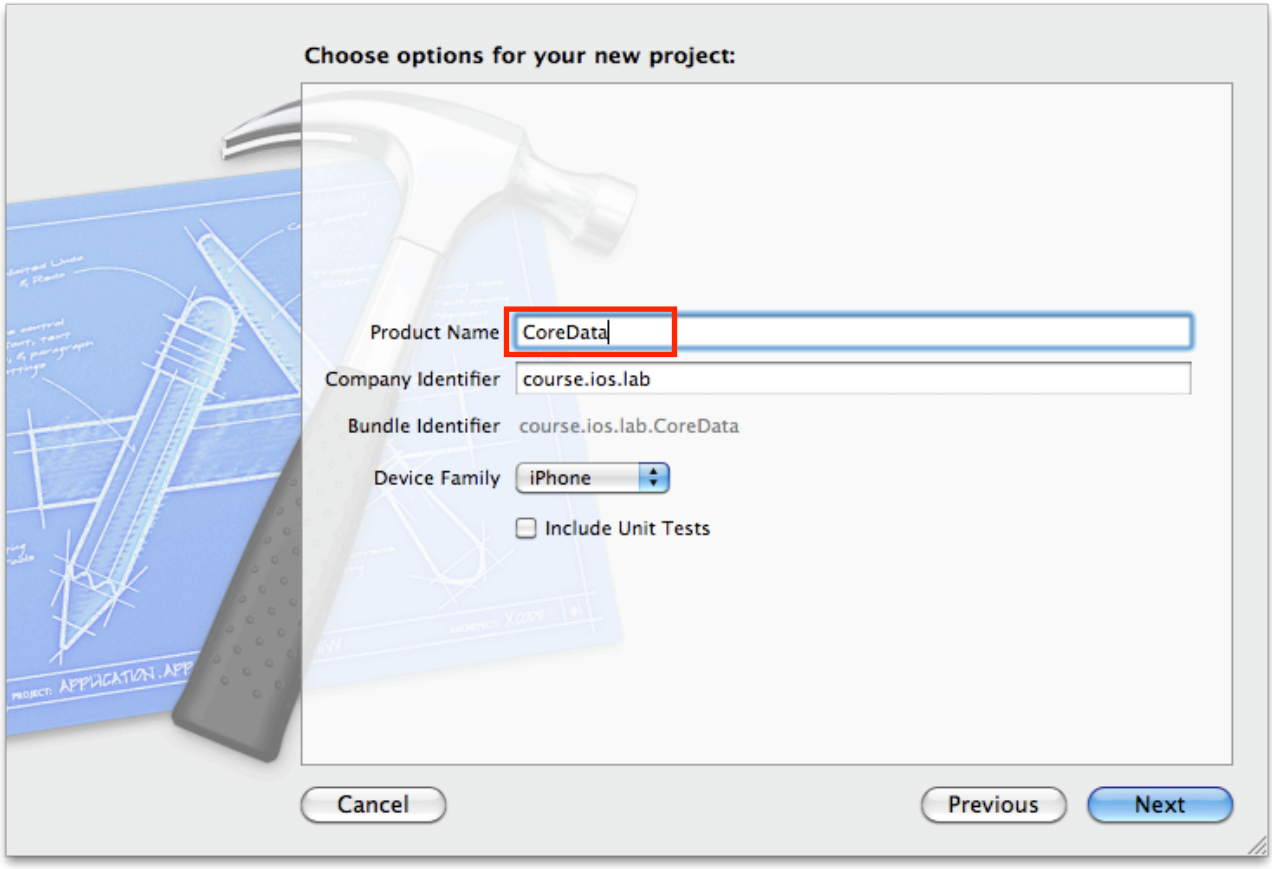
Lab CoreData

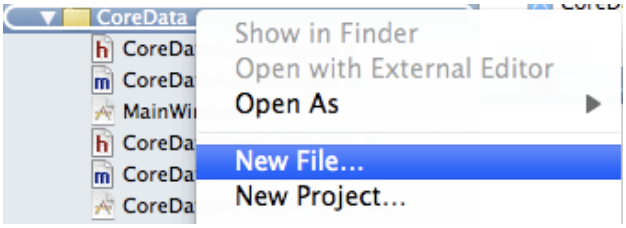
Step 1. 在 File>New>New File 開啓一個新的project, 選擇 View-based application



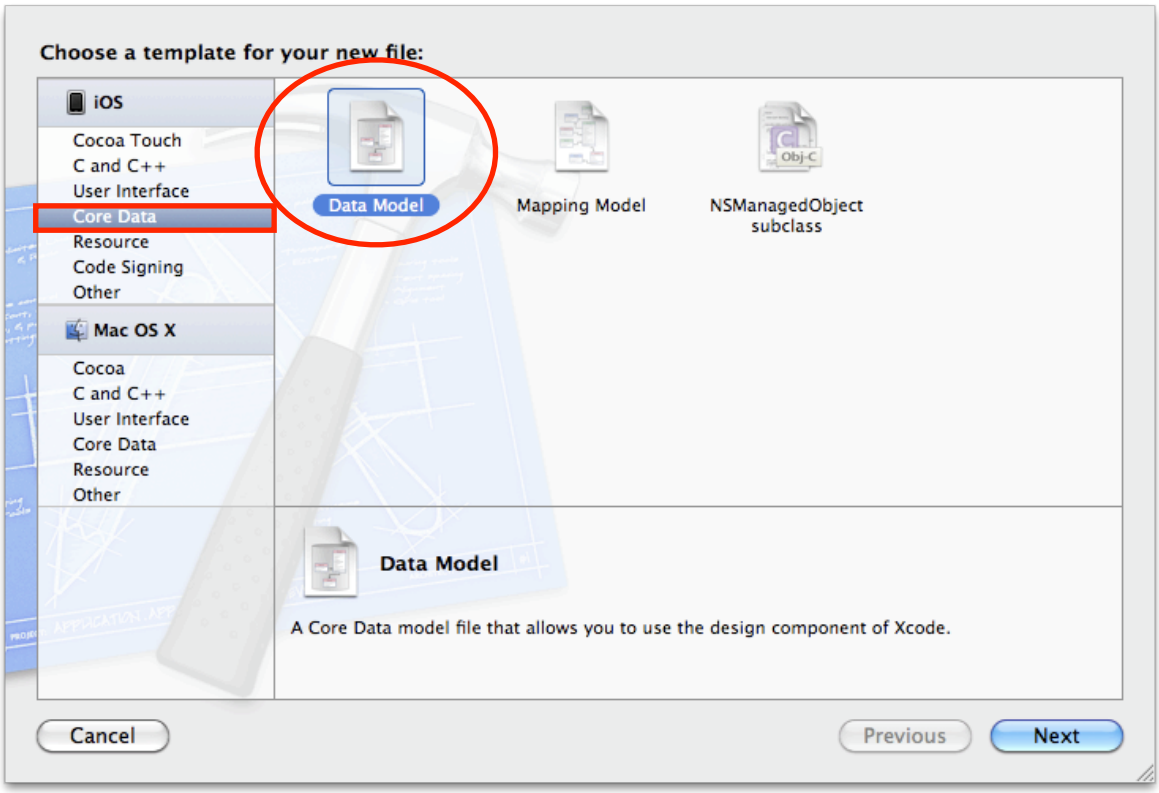
將project命名為 **CoreData**



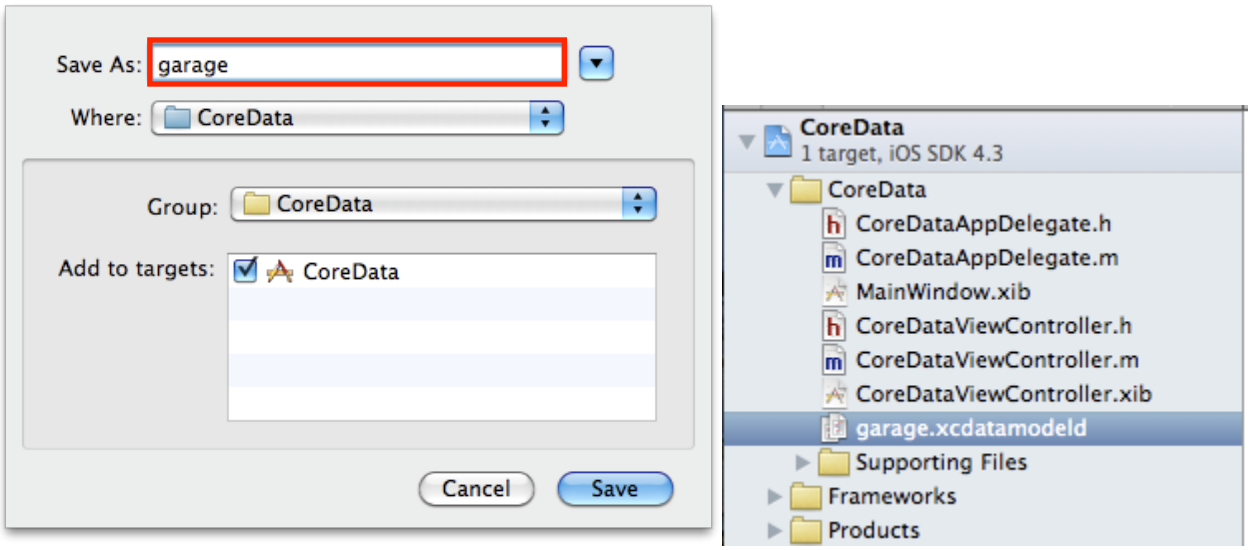
Step 2. 在 CoreData 資料夾點右鍵>New File

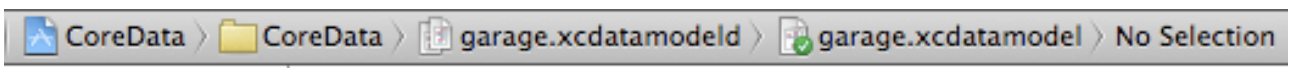


選擇 iOS 的 Core Data目錄裡的 Data Model

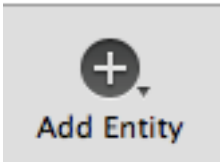


取名為 garage, 就可以看到 一個 Core Data package garage.xcdatamodeld

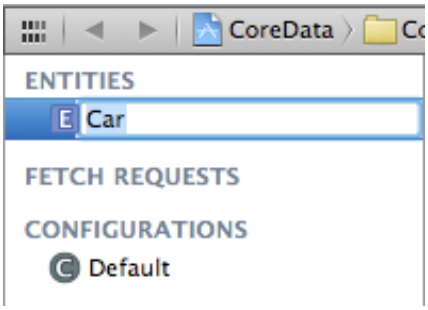




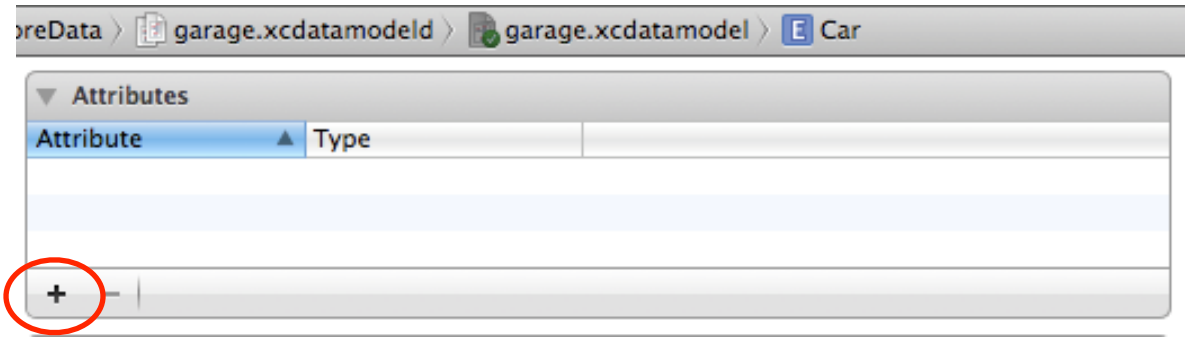
Step 3. 點選 garage.xcdatamodeld 在中間視窗最下面點選 Add Entity



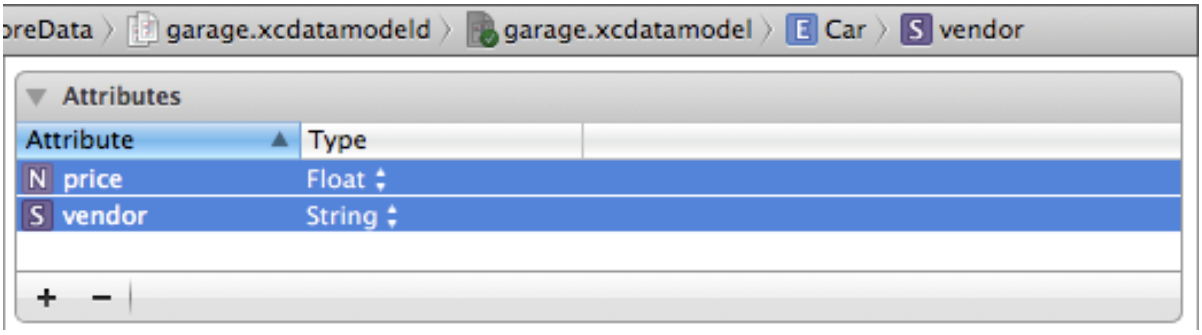
則會出現一個新的Entity在ENTITIES這欄.我們將此命名為 **Car**



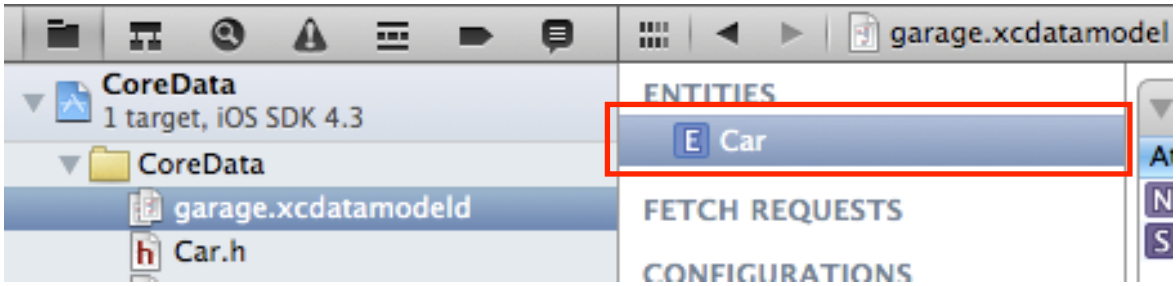
Step 4. 點選剛剛新增的Car這個Entity, 在Attribute 這欄點下面的加號來增加2個 attribute



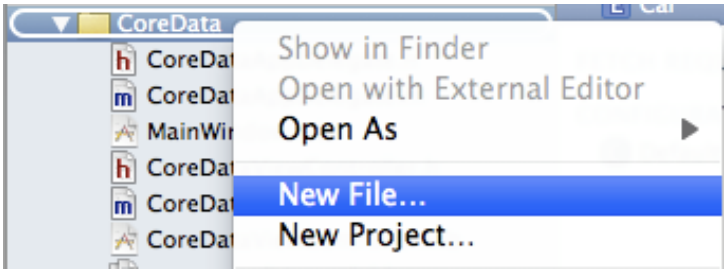
分別命名為 price, vendor. 點選 price, 在右邊的 type 選單中, 將 price的 type選為 float, 而將 vendor 的 type選為 String



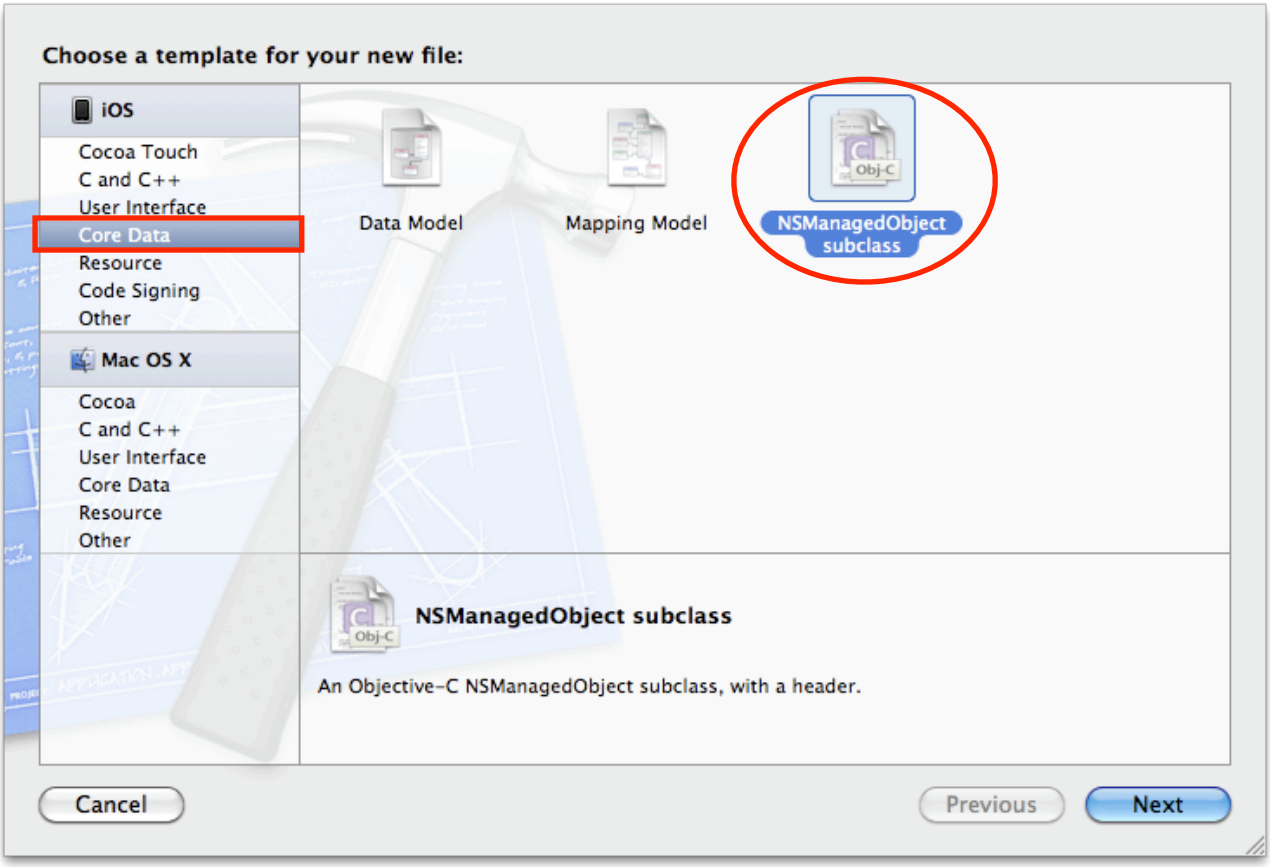
Step 5-1. 先開啟 garage.xcdatamodeld 裡的 garage.xcdatamodel 並點選 Car 這個 Entity



Step 5-2. 在左邊Project Navigator視窗裡在CoreData資料夾上點右鍵,選擇New File

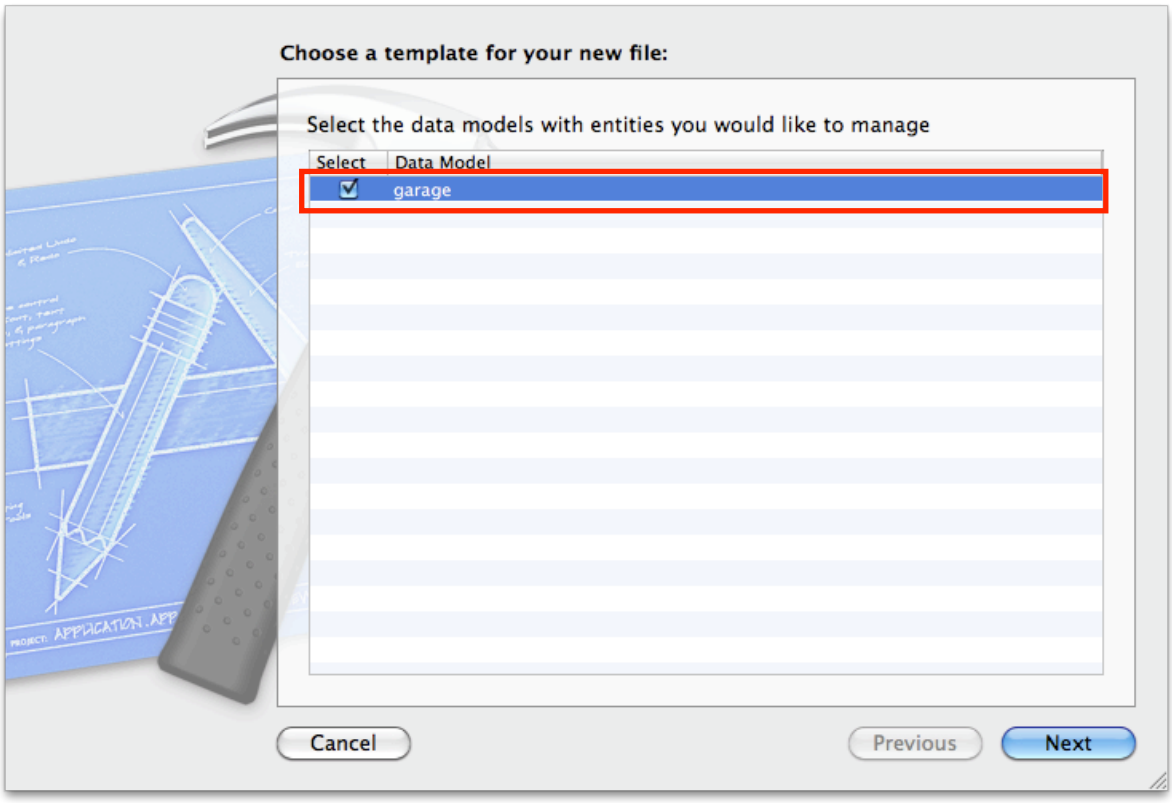


Step 5-3. 選擇iOS裡的Core Data目錄下的 NSManagedObject subclass



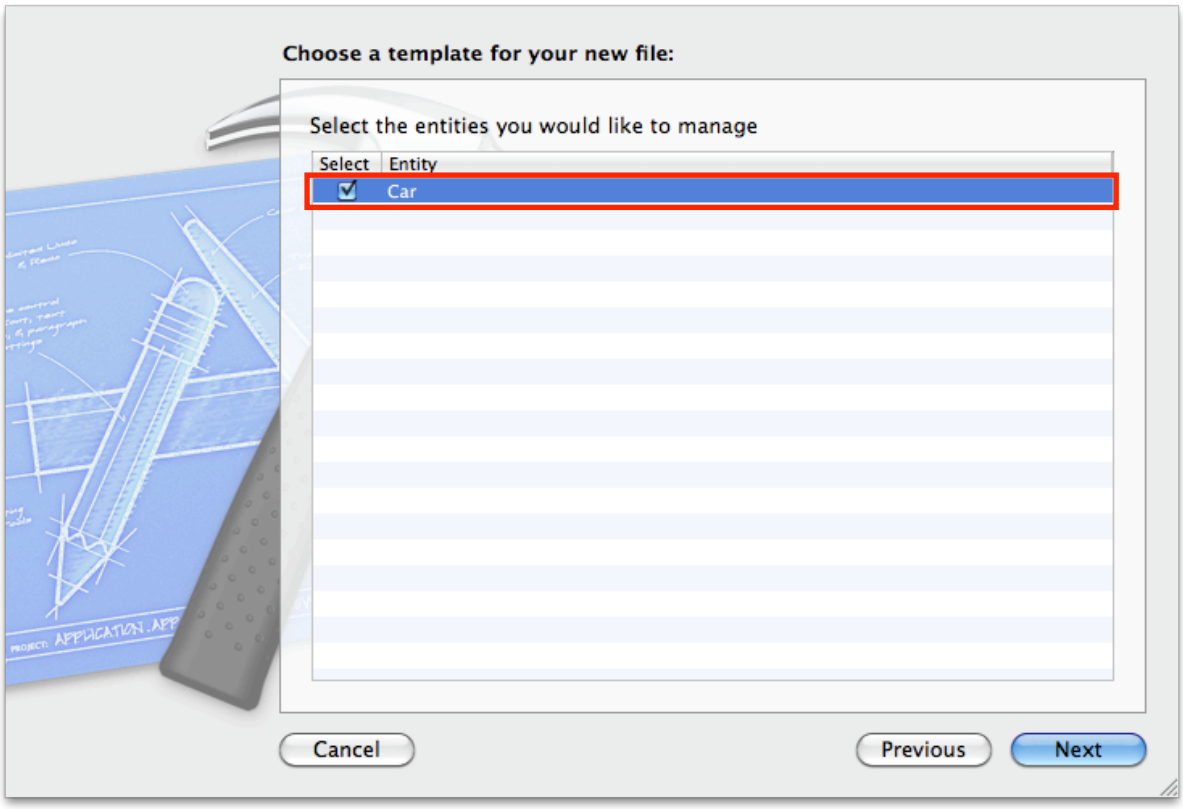
Step 5-4. Template Data Model 選擇我們剛剛新增的 Data Model **Garage**

(若在 Step 5-1 沒有開啓 garage.xcdatamodel 會需要此步驟)

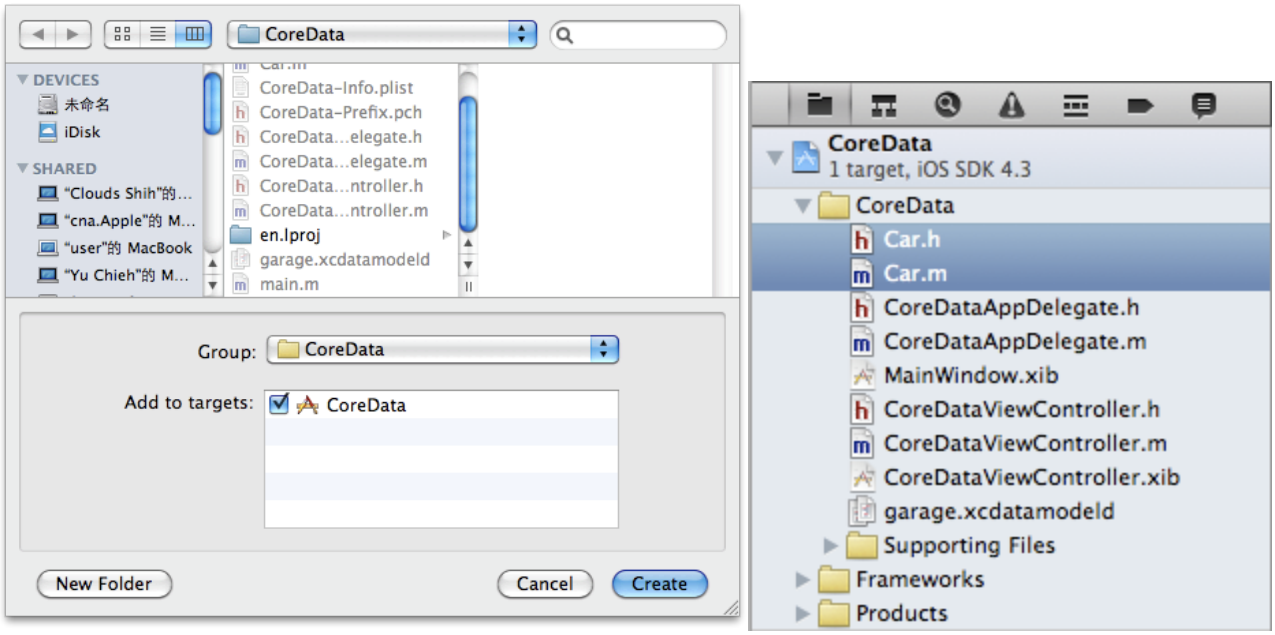


Template Entity 選擇我們剛剛新增的Entity **Car**

(若在 Step 5-1 沒有點選 Car 這個 Entity 會需要此步驟)



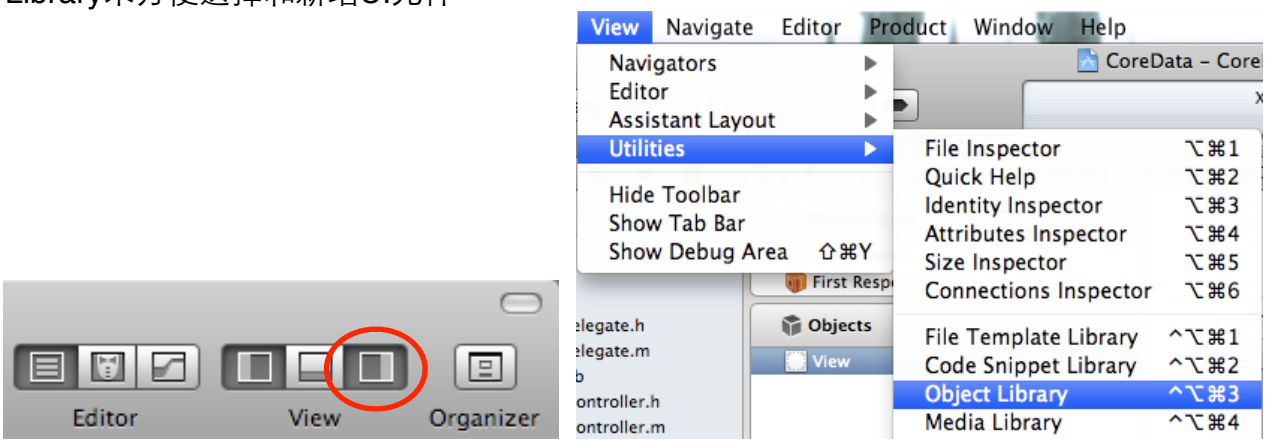
存檔後在Project中便加入了Managed object class Car.m 以及 Car.h



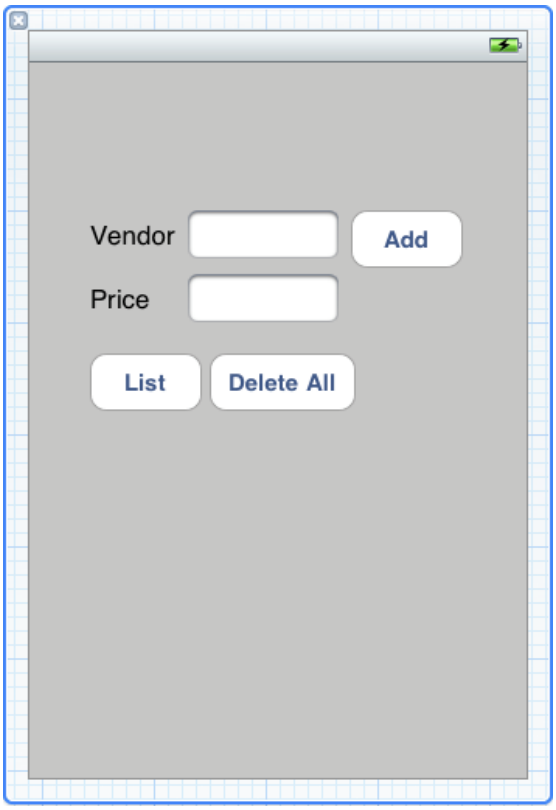
Step 6. 在左邊Project Navigator開啓 CoreDataViewController.h, 先 `#import "Car.h"`, 接著加入三個 button, Textfield, NSManagedObjectContext 物 件, 然後定義三個按鈕的 action.

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

@interface CoreDataViewController : UIViewController {
    IBOutlet UIButton *buttonAdd;
    IBOutlet UIButton *buttonList;
    IBOutlet UIButton *buttonDelete;
    IBOutlet UITextField *carName;
    IBOutlet UITextField *carPrice;
    NSManagedObjectContext *managedObjectContext;
}
-(IBAction) addCar;
-(IBAction) listCar;
-(IBAction) deleteAllCar;
@end
```

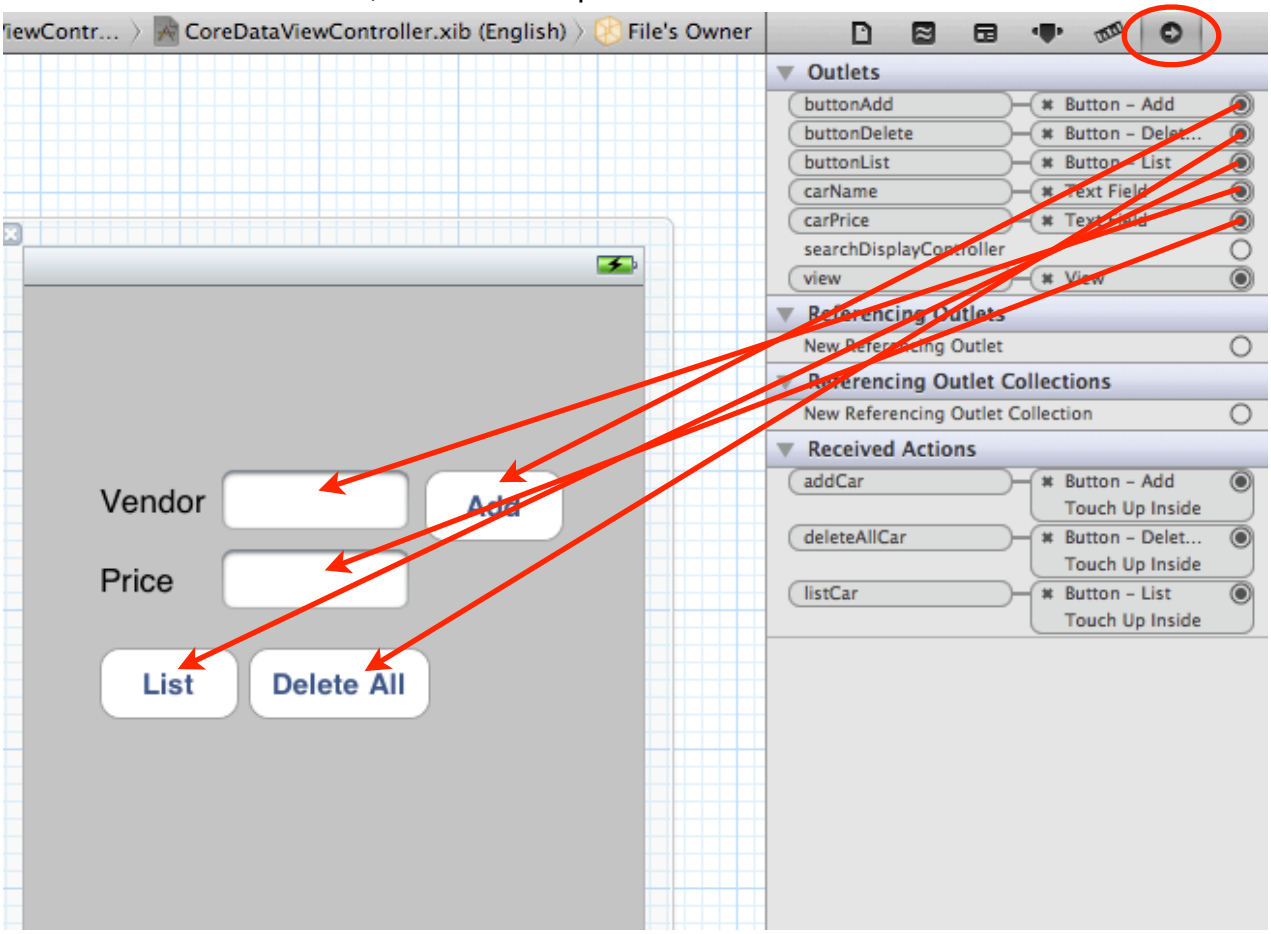
Step 7. 開啓 CoreDataViewController.xib, 在Xcode視窗右上角選擇顯示視窗點下View右邊按鍵,或是View > Utilities 裡選擇任意選擇一個選項即會打開這個視窗分頁,在此選擇Object Library來方便選擇和新增UI元件



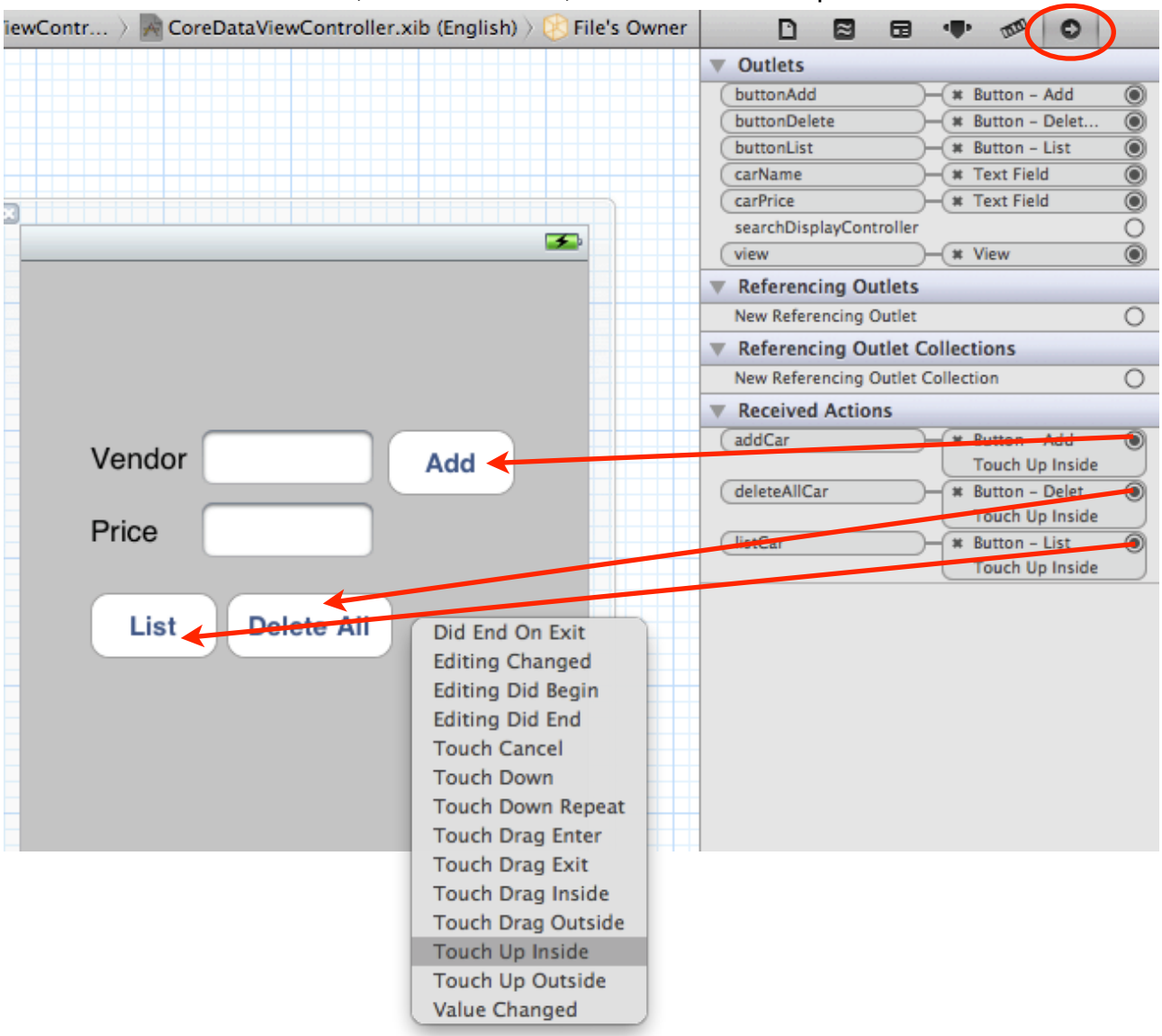
接著我們使用Object Library, 在View擺上2個Label, 2個TextField, 和 3個Button (注意這幾個元件盡量放在View的上半部,避免Runtime輸入時被虛擬鍵盤蓋住)



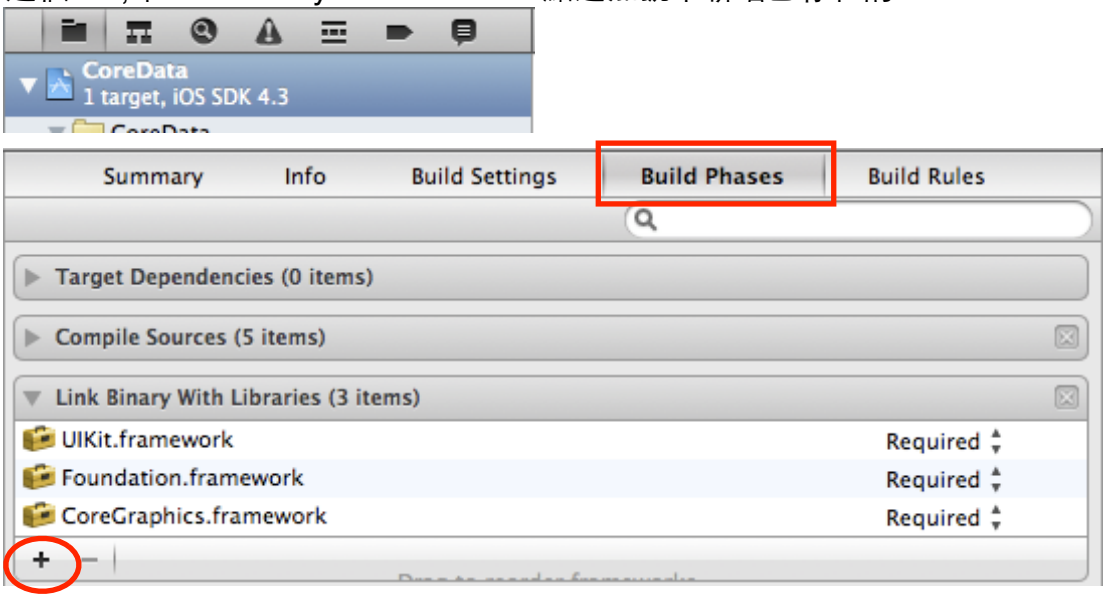
Step 8. 點選File's Owner ,開啓Connections Inspector, 將剛才在CoreDataViewController.h中定義的 buttonAdd 連到 Add, buttonList 連到 List, buttonDelete 連到 Delet All, carName 連到 Vendor 旁的 textfield, carPrice 連到price旁的 textfield.



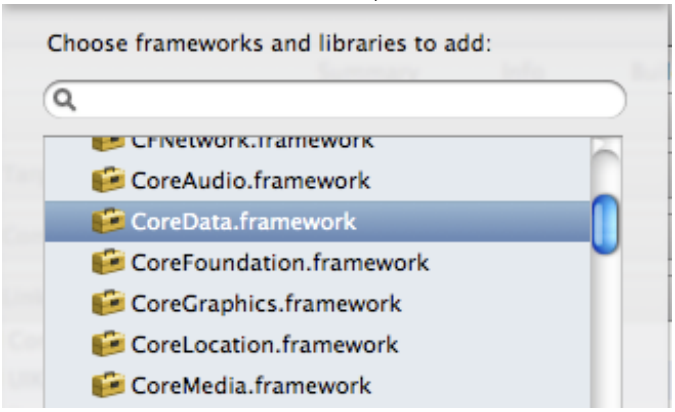
Step 9. 將三個IBAction和View上的三個Buton的Event作連結, addCar 連到 Add, deleteAllCar 連到 Delete All, listCar連到 List, Event選擇Touch Up Inside



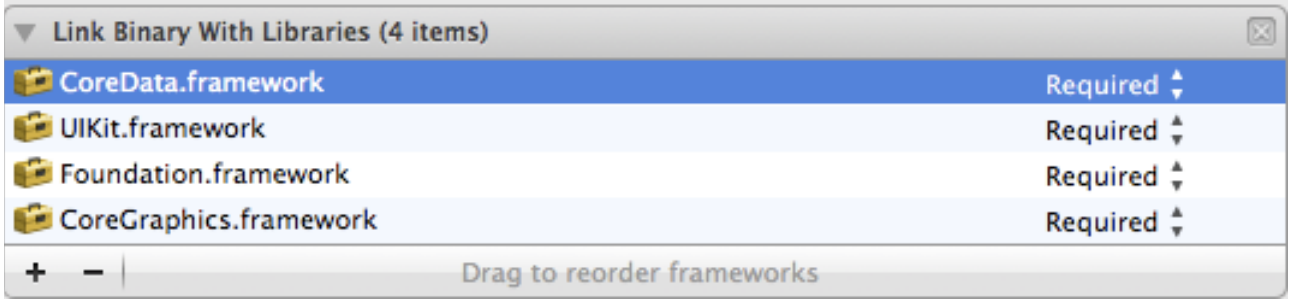
Step 10. 加入compile時會需要的 framework, 點 CoreData這個Project檔, 選擇Build Phases 這個Tab, 在Link Binary With Libraries 點選加號來新增已存在的framework



選擇 CoreData.framework, 點Add



CoreData.framework 就出現在Link Binary With Libraries欄中



Step 11. 開啓 CoreDataViewController.m, 找到並把 ViewDidLoad 的Mark去掉, 加入下面的程式.

```
- (void)viewDidLoad
{
    [super viewDidLoad];

    NSArray *paths = NSSearchPathForDirectoriesInDomains
(NSDocumentDirectory, NSUserDomainMask, YES);
    NSString *basePath = ([paths count] > 0) ? [paths objectAtIndex:0]:
nil;
    NSURL *storeURL = [NSURL fileURLWithPath:[basePath
stringByAppendingPathComponent:@"Car.sqlite"]];
    NSPersistentStoreCoordinator *persistentCoordinator =
    [[NSPersistentStoreCoordinator alloc] initWithManagedObjectModel:
[NSManagedObjectModel mergedModelFromBundles:nil]];
    NSError *error = nil; // a must

    if(![persistentCoordinator
addPersistentStoreWithType:NSSQLiteStoreType configuration:nil
URL:storeURL options:nil error:&error])
    {
        NSLog(@"error loading persistent store....");
    }
    managedObjectContext = [[NSManagedObjectContext alloc] init];
    [managedObjectContext
setPersistentStoreCoordinator:persistentCoordinator];
    [error release];

}
```

Step 12. 接著實作 addCar 這個action

```
-(IBAction) addCar {

    Car *newOne = [NSEntityDescription
insertNewObjectForEntityForName:@"Car"
                   inManagedObjectContext:managedObjectContext];
    newOne.vendor = carName.text;
    newOne.price = [NSNumber numberWithInt:[carPrice.text floatValue]];

    NSError *error = nil;

    if(![managedObjectContext save:&error]) {
        NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
        [error release];
    }

}
```

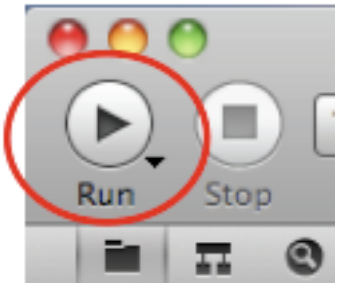
Step 13. 接著實作 listCar 這個action

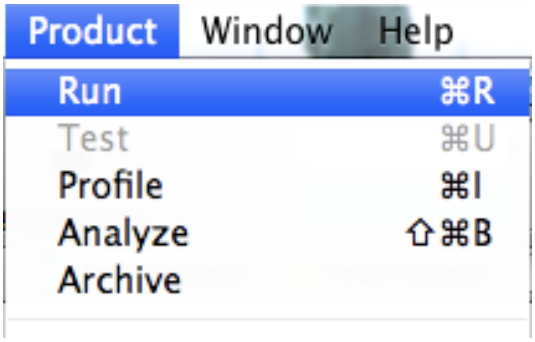
```
-(IBAction) listCar {

    NSFetchRequest *request = [[NSFetchRequest alloc] init];
    NSEntityDescription *entity = [NSEntityDescription
entityForName:@"Car" inManagedObjectContext:managedObjectContext];
    [request setEntity:entity];
    NSPredicate *predicate = [NSPredicate predicateWithFormat:@"price >
%@", [NSNumber numberWithFloat:(float)500]];
    [request setPredicate:predicate];

    NSError *error = nil;
    NSArray *fetchResults = [managedObjectContext
executeFetchRequest:request error:&error];

    if(fetchResults == nil){
        NSLog(@"an error occured");
    }
    else{
        NSLog(@"List Car");
        for(Car *car in fetchResults) {
            NSLog(@"got car with name %@ and price
%@",car.vendor,car.price);
        }
    }

}
```

Step 14. 接著實作 deleteAllCar 這個action

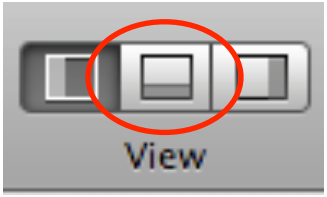
```
-(IBAction) deleteAllCar {

    NSFetchRequest *request = [[NSFetchRequest alloc] init];
    NSEntityDescription *entity = [NSEntityDescription
entityForName:@"Car" inManagedObjectContext:managedObjectContext];
    [request setEntity:entity];

    NSError *error = nil;
    NSArray *fetchResults = [managedObjectContext
executeFetchRequest:request error:&error];

    if(fetchResults == nil) {
        NSLog(@"an error occured");
    }
    else{
        for(NSManagedObject *object in fetchResults) {
            [managedObjectContext deleteObject:object];
        }
        if(![managedObjectContext save:&error]){
            NSLog(@"Unresolved error %@, %@", error, [error userInfo]);
            [error release];
        }
    }

}
```

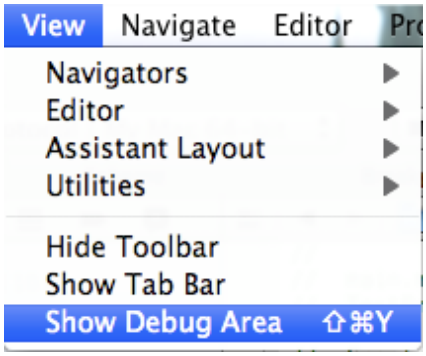
Step 15. Run (⌘+R)
在Xcode主頁左上角按下Run, 或是在Product > Run, 即開始Build code並執行



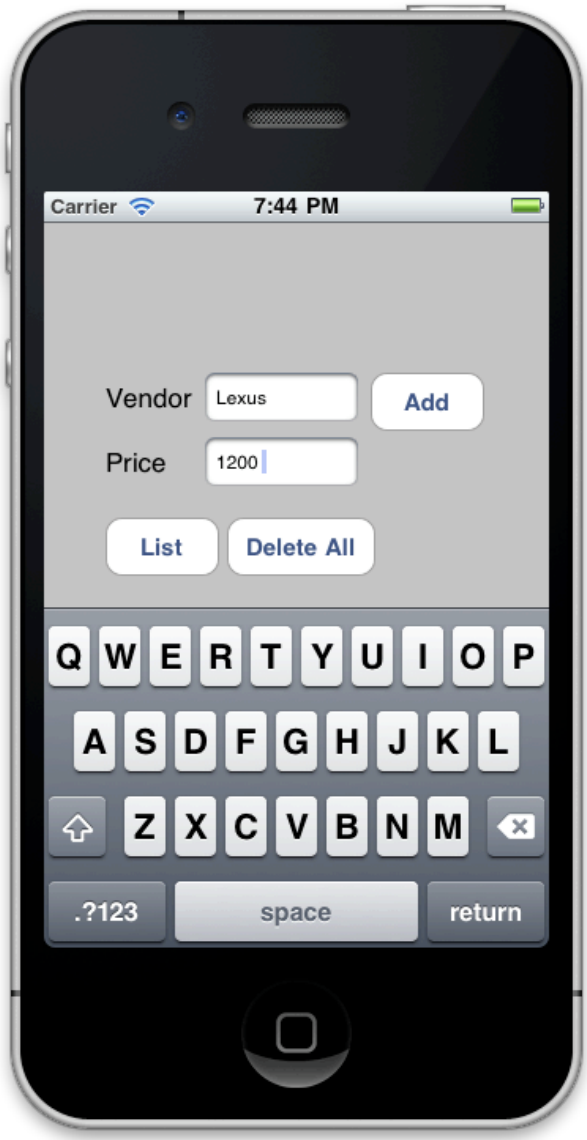


之後會自動開啓console, 沒開啓的話在右上角View點擊中間的Button, 或是選擇View > Show Debug Area 來開啓





我們輸入兩組資訊,vendor (VW) price(1000) -> Add, vendor(Lexus) price (1200) -> Add 接著按 list, 把所有資訊傳出來看看



All Output Clear

```
GNU gdb 6.3.50-20050815 (Apple version gdb-1518) (Sat Feb 12 02:52:12 UTC 2011)
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "x86_64-apple-darwin".Attaching to process 4891.
2011-04-03 19:48:31.356 CoreData[4891:207] List Car
2011-04-03 19:48:31.360 CoreData[4891:207] got car with name VW and price 1000
2011-04-03 19:48:31.361 CoreData[4891:207] got car with name Lexus and price 1200
```