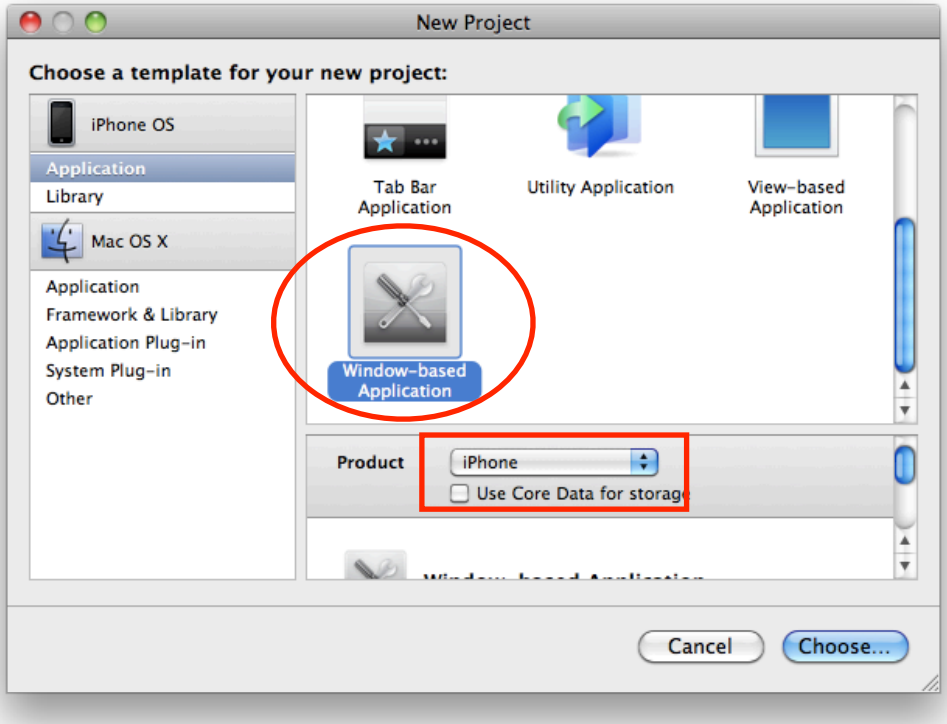
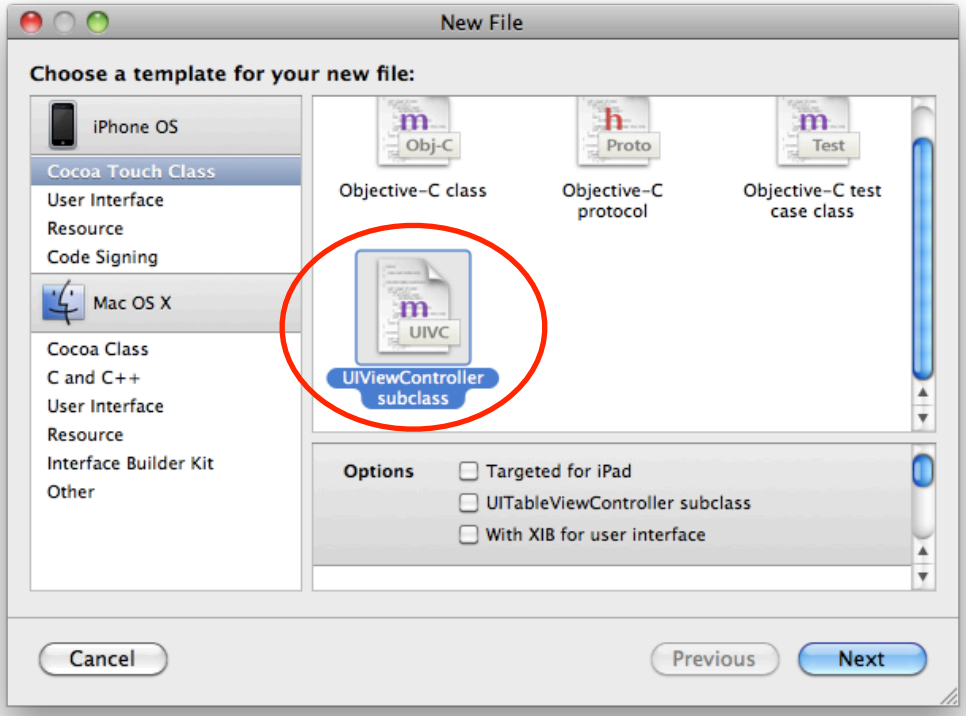
Lab codeUIViewController

Step1. 在File開啓一個新的project, 因為我們要實作viewController ,這次我們選擇 window-based application, 將project取名為 codeUIViewController



Step2. 在File裡選擇New File,在Cocoa Touch Classes裡選擇UIViewController subclass,命名為MyViewController,在此先不選擇With XIB for user interface,勾選Also create “MyViewController.h”來一起建立header檔



Step 3. classes > MyViewController.h , 我們定義一個label 一個 button, 以及一個給 button 用的 action, myAction

```
@interface MyViewController : UIViewController {
    UIButton* myButton;
    UILabel* myLabel;
}

-(void) myAction;

@end
```

Step4. classes > MyViewController.m, 實作 constructor, destructor, 以及 myAction, destructor是在系統產生的 `-(void)dealloc{}` 加入程式碼.

```
-(id) init{
    if(self = [super init])
    { }
    return self;
}

-(void) myAction
{
    [myLabel setText:@"Hello MyViewController"];
}


- (void)dealloc {
    [myLabel release];
    [myButton release];
    [super dealloc];
}
```

Step 5. classes > MyViewController.m , 找到 loadView. 我們在裡面實作UIview, UIlabel 以及 UIButton 物件

```
// Implement loadView to create a view hierarchy programmatically, without using a nib.
- (void)loadView {
    UIView* myView=[[UIView alloc] initWithFrame:[UIScreen mainScreen].applicationFrame ] ;
    myButton =[UIButton buttonWithType:UIButtonTypeRoundedRect];
    myButton.frame = CGRectMake(110, 150, 100, 50);
    myLabel = [[UILabel alloc] initWithFrame:CGRectMake(100, 100, 300, 50)];
    [myButton setTitle:@"push" forState:UIControlStateNormal];
    [myLabel setText:@"Label"];
    [myButton addTarget:self action:@selector(myAction) forControlEvents:UIControlEventTouchUpInside];
    [myView addSubview:myLabel];
    [myView addSubview:myButton];
    self.view = myView;
    [myView release];
}
```

```
UIView* myView=[[UIView alloc] initWithFrame:[UIScreen mainScreen].applicationFrame];
```

這一段是開一個 UIView 形態的 view, 當作最底層的 view, 我們後面會將這個view傳給 UIController 管理.

```
myButton =[UIButton buttonWithType:UIButtonTypeRoundedRect];
```

將 myButton 初始化, 並且將它的 type 設成前面實驗都用到的 UIButtonTypeRoundedRect.
更多的形態,請查 Help > Documentation, 找UIButtonType

```
myButton.frame = CGRectMake(110, 150, 100, 50);
```

這一段是將myButton在myView上的位置和大小定義出來

```
myLabel = [[UILabel alloc] initWithFrame:CGRectMake(100, 100, 300, 50)];
```

直接將 myLabel初始化, 而且直接用 initWithFrame 將它的位置大小定義出來

```
[myButton setTitle:@"push" forState:UIControlStateNormal];
[myLabel setText:@"Label”];
```

設定myButton 以及 myLabel上的文字

```
[myButton addTarget:self action:@selector(myAction)
forControlEvents:UIControlEventTouchUpInside];
```

我們將myAction與myButton的 touch down動作連結起來.

```
[myView addSubview:myLabel];
[myView addSubview:myButton];
```

接著我們把myLabel以及myButton加到myView裡

```
self.view = myView;
[myView release];
```

我們將 myView 傳給這個 UIViewController管理,然後就可以把 myView release掉了.

Step 6. classes > codeViewControllerAppDelegate.m ,首先

```
#import "MyViewController.h"
```

接著在 `application didFinishLaunchingWithOptions:`中

```
#import "codeUIViewControllerAppDelegate.h"
#import "MyViewController.h"

@implementation codeUIViewControllerAppDelegate

@synthesize window;


#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions

    // Override point for customization after application launch.
    MyViewController* myViewController = [[MyViewController alloc] init];
    [window addSubview:myViewController.view];
    [window makeKeyAndVisible];

    return YES;
}
```

```
MyViewController* myViewController = [[MyViewController alloc] init];
```

我們首先 initialize 一個 MyViewController形態的 myViewController

```
[window addSubview:myViewController.view];
```

接著我們將 myViewController 的 view 交給 window 管理.

Step 7. build and GO, 當我們 按下 push的時候, label就被設定成 Hello MyViewController