

IOS 学习之 UINavigationController 详解与使用(一)添加 UIBarButtonItem

1、UINavigationController 导航控制器如何使用

UINavigationController 可以翻译为导航控制器，在 IOS 里经常用到。

我们看看它的如何使用：

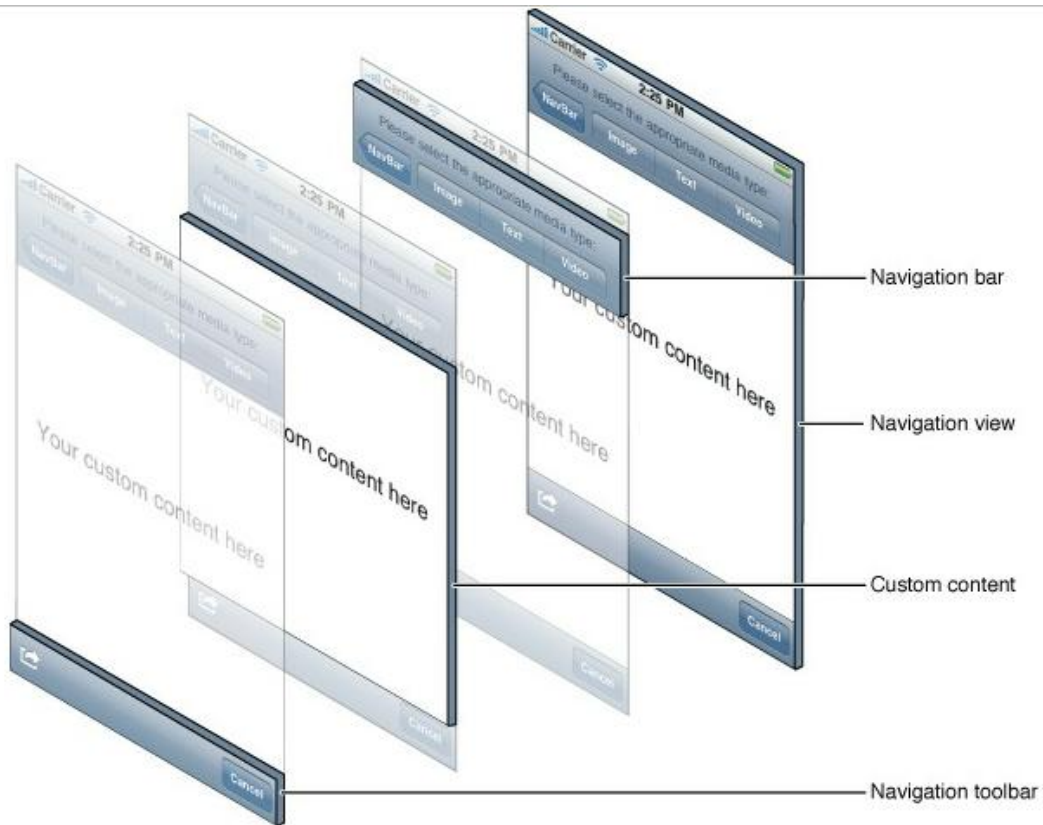
下面的图显示了导航控制器的流程。最左侧是根视图，当用户点击其中的 **General** 项时，**General** 视图会滑入屏幕；当用户继续点击 **Auto-Lock** 项时，**Auto-Lock** 视图将滑入屏幕。相应地，在对象管理上，导航控制器使用了导航堆栈。根视图控制器在堆栈最底层，接下来入栈的是 **General** 视图控制器和 **Auto-Lock** 视图控制器。可以调用 `pushViewControllerAnimated:` 方法将视图控制器推入栈顶，也可以调用 `popViewControllerAnimated:` 方法将视图控制器弹出堆栈。



上图来自苹果官网。

2、UINavigationController 的结构组成

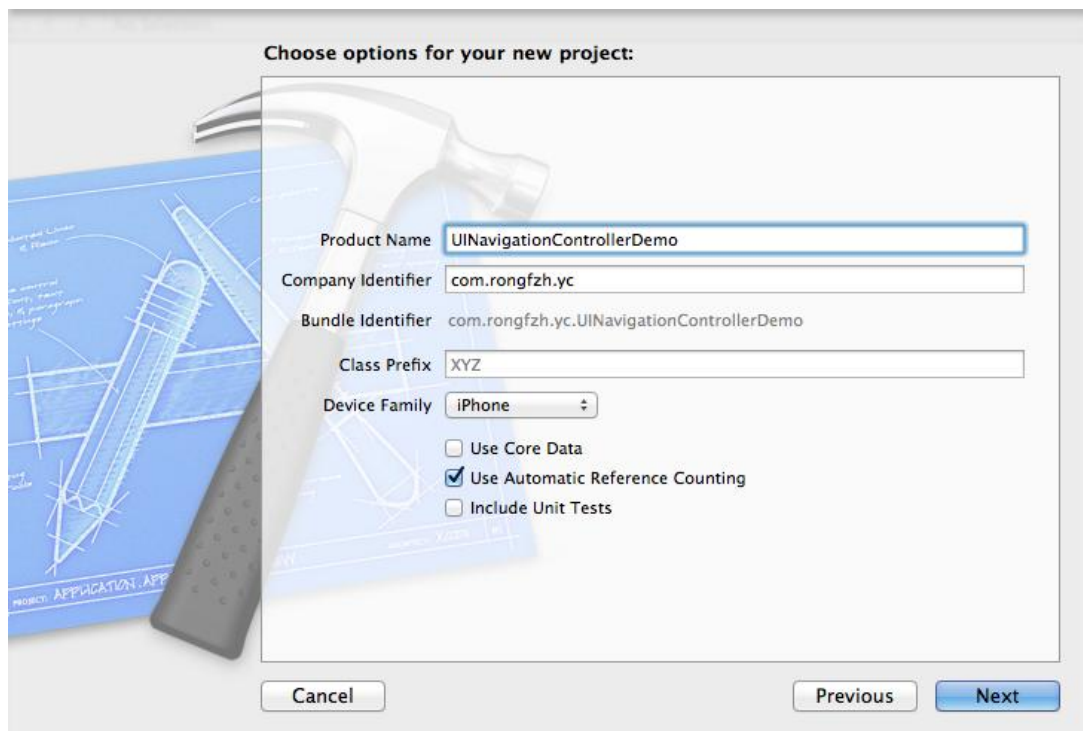
看下图，UINavigationController 有 Navigation bar ,Navigation View ,Navigation toolbar 等组成。



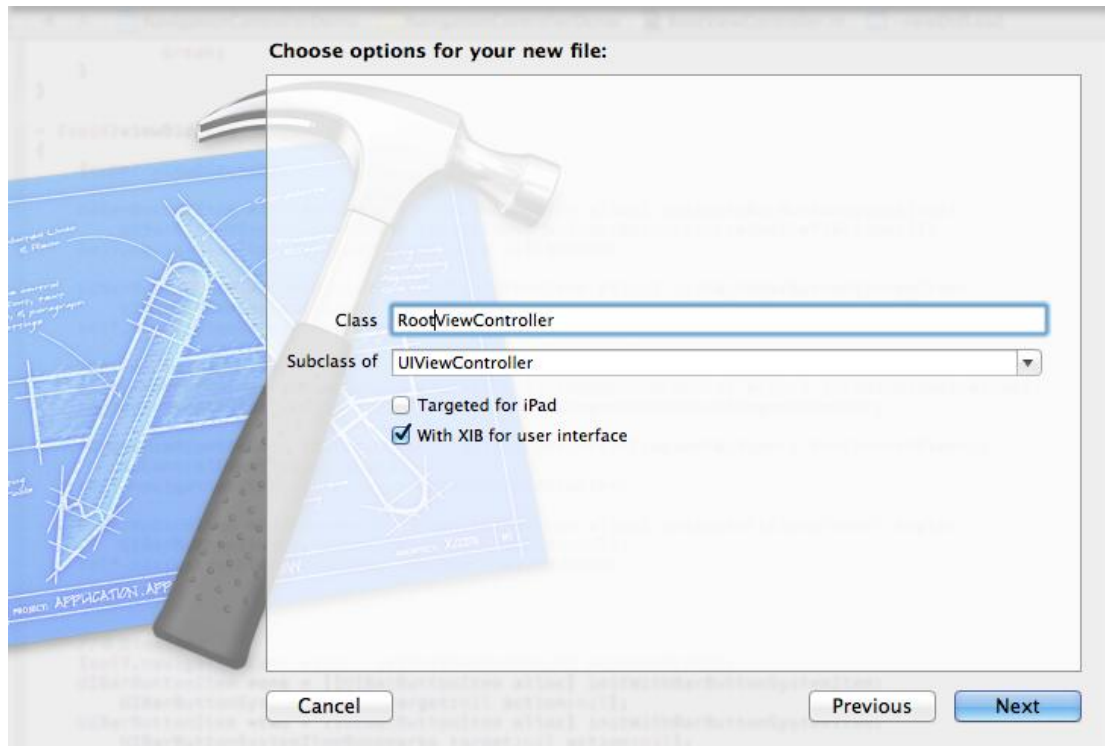
现在我们建立一个例子，看看如何使用 UINavigationController

3、新建一个项目

命名为 UINavigationControllerDemo,为了更好理解 UINavigationController, 我们选择 Empty Application 模板



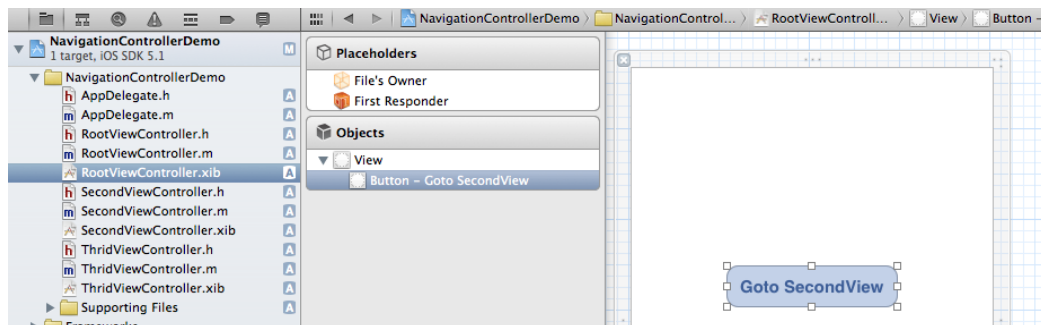
4、创建一个 View Controller，命名为 RootViewController：依次选择 File——New——New File，默认勾上 With XIB for user interface.



选择正确位置创建完成，这时项目里多了三个文件，分别是

RootViewController.h RootViewController.m RootViewController.xib 文件。

打开 RootViewController.xib，添加一个按钮控件，按钮 Button 改成：Goto SecondView，为跳转做准备



5、打开 AppDelegate.h，向其中添加属性：

[cpp] view plaincopy

```
1. @property (strong, nonatomic) UINavigationController *navController;
```

添加后 AppDelegate.h 文件代码如下：

[cpp] view plaincopy

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

本文由吉林白癜风医院 <http://tf463.com/> 收集，转载请注明出处

```
2.
3. @class ViewController;
4.
5. @interface AppDelegate : UIResponder <UIApplicationDelegate>
6.
7. @property (strong, nonatomic) UIWindow *window;
8.
9. @property (strong, nonatomic) ViewController *viewController;
10.
11. @property (strong, nonatomic) UINavigationController *navController;
12.
13. @end
```

6、在 AppDelegate.m 文件的 didFinishLaunchingWithOptions 方法中创建添加 navController, RootViewController 视图。

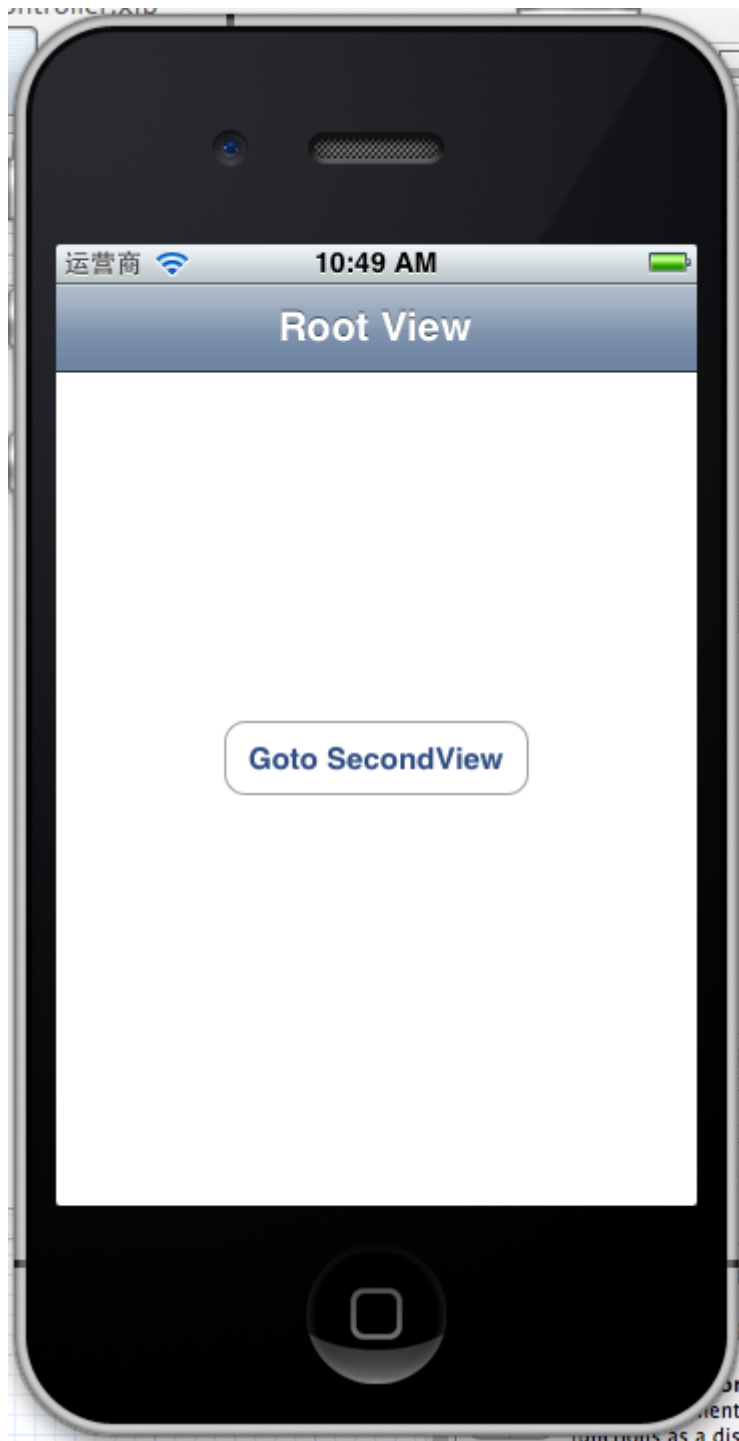
[cpp] view plaincopy

```
1. - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
2. {
3.     self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]];
4.     RootViewController *rootView = [[RootViewController alloc] init];
5.     rootView.title = @"Root View";
6.
7.     self.navController = [[UINavigationController alloc] init];
8.     [self.navController pushViewController:rootView animated:YES];
9.     [self.window addSubview:self.navController.view];
10.    [self.window makeKeyAndVisible];
11.    return YES;
12. }
```

给 rootView 的 title 命名为 Root View, 好识别 View 直接的切换关系。用 pushViewController 把 rootView 加入到 navController 的视图栈中。

7、现在 Root 视图添加完成

看看效果：



现在还没有 Navigation bar 。只有 title。

8、添加 UIBarButtonItem

UIBarButtonItem 分左右 UIBarButtonItem。我们把左右的都添加上去。

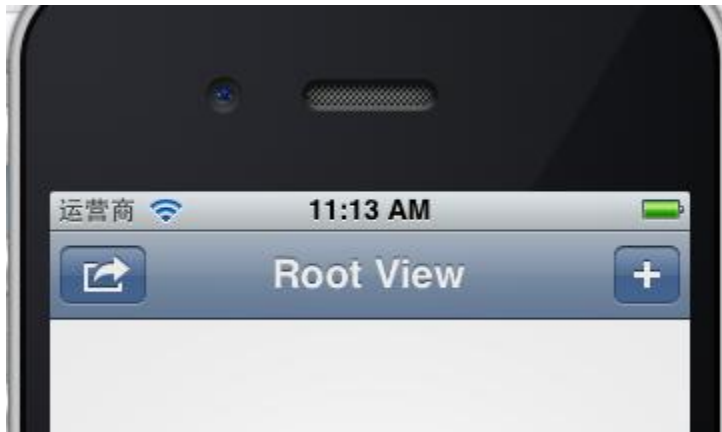
在 RootViewController.m 中添加代码如下：

[[cpp](#)] [view plaincopy](#)

```
1. - (void)viewDidLoad
2. {
```

```
3. [super viewDidLoad];
4.
5. UIBarButtonItem *leftButton = [[UIBarButtonItem alloc] initWithBarButton
    SystemItem:UIBarButtonSystemItemAction target:self action:@selector(selectLe
    ftAction:)];
6. self.navigationItem.leftBarButtonItem = leftButton;
7.
8. UIBarButtonItem *rightButton = [[UIBarButtonItem alloc] initWithBarButto
    nSystemItem:UIBarButtonSystemItemAdd target:self action:@selector(selectRig
    htAction:)];
9. self.navigationItem.rightBarButtonItem = rightButton;
```

这样添加了 UIBarButtonItem 了，效果如下：



















这里重点介绍下

UIBarButtonItem *leftButton =

```
[[UIBarButtonItem alloc] initWithBarButtonSystemItem:UIBarButtonSystemItemActiontarget:
selfaction:@selector(selectLeftAction:);
```

UIBarButtonItemAction 的风格，这是系统自带的按钮风格，看下图，你不用一个个试验，你也知道想用那个 item，如下图：

标签	效果	标签	效果
UIBarButtonItemSystemItemAction		UIBarButtonItemSystemItemPause	
UIBarButtonItemSystemItemAdd		UIBarButtonItemSystemItemPlay	
UIBarButtonItemSystemItemBookmarks		UIBarButtonItemSystemItemRedo	
UIBarButtonItemSystemItemCamera		UIBarButtonItemSystemItemRefresh	
UIBarButtonItemSystemItemCancel		UIBarButtonItemSystemItemReply	
UIBarButtonItemSystemItemCompose		UIBarButtonItemSystemItemRewind	
UIBarButtonItemSystemItemDone		UIBarButtonItemSystemItemSave	
UIBarButtonItemSystemItemEdit		UIBarButtonItemSystemItemSearch	
UIBarButtonItemSystemItemFastForward		UIBarButtonItemSystemItemStop	
UIBarButtonItemSystemItemOrganize		UIBarButtonItemSystemItemTrash	
UIBarButtonItemSystemItemPageCurl		UIBarButtonItemSystemItemUndo	

9、响应 UIBarButtonItem 的事件的实现

我们在 `action:@selector(selectLeftAction:);`
action 添加了 `selectLeftAction` 和 `selectRightAction`
在 `RootViewController.m` 文件中添加代码实现：

[cpp] [view plaincopy](#)

```
1. -(void)selectLeftAction:(id)sender
2. {
3.     UIAlertView *alter = [[UIAlertView alloc] initWithTitle:@"提示
    " message:@"你点击了导航栏左按钮" delegate:self cancelButtonTitle:@"确定
    " otherButtonTitles:nil, nil];
4.     [alter show];
5. }
6.
7. -(void)selectRightAction:(id)sender
8. {
```

```
9.     UIAlertView *alter = [[UIAlertView alloc] initWithTitle:@"提示  
    " message:@"你点击了导航栏右按钮" delegate:self cancelButtonTitle:@"确定  
    " otherButtonTitles:nil, nil];  
10.    [alter show];  
11. }
```

这样在点击左右的 UIBarButtonItem 时，弹出提示：



这篇先讲添加 UIBarButtonItem，下篇讲解页面跳转和添加 UISegmentedControl