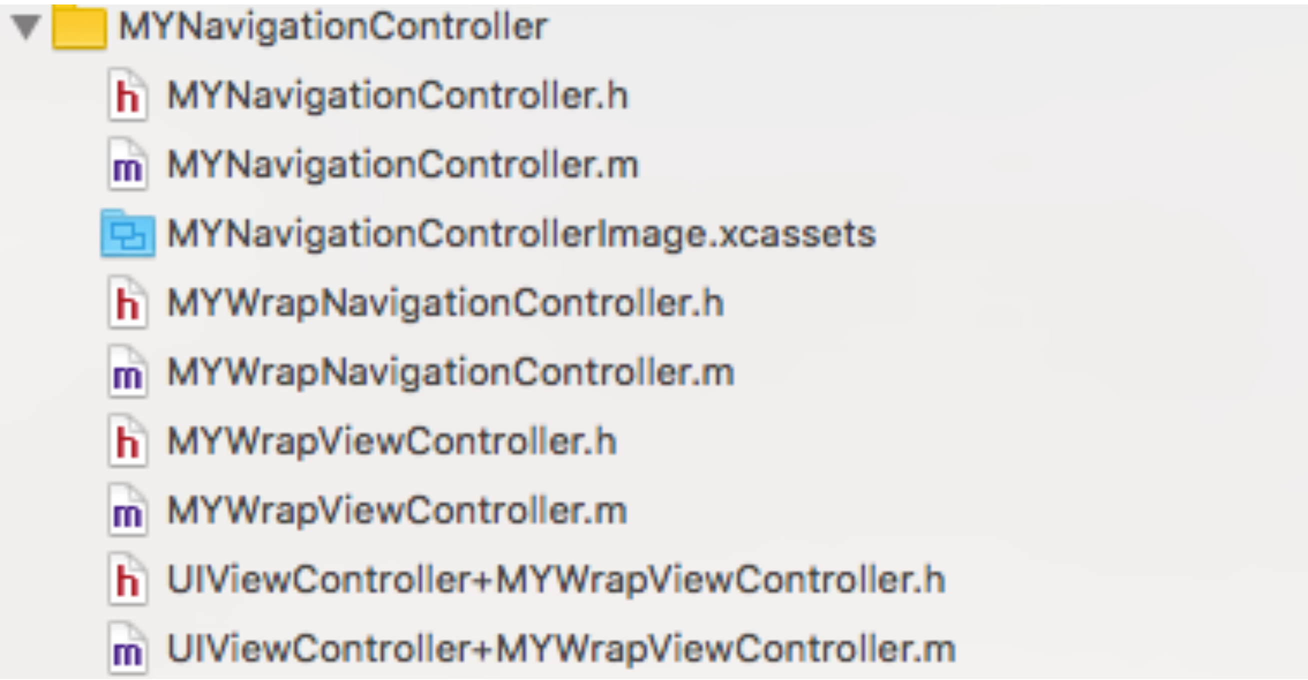
MYNavigationController 封装

1. 简述：利用 UIViewAnimatedTransition 可以实现视觉上的独立 NavigationBar 的转场动画效果，而本次封装能让每一个 viewController 都有一个自己的拥有独立 navigationBar 的 NavigationController。

2. 实现：

(1) 目录结构



MYNavigationController

h MYNavigationController.h

m MYNavigationController.m

MYNavigationControllerImage.xcassets

h MYWrapNavigationController.h

m MYWrapNavigationController.m

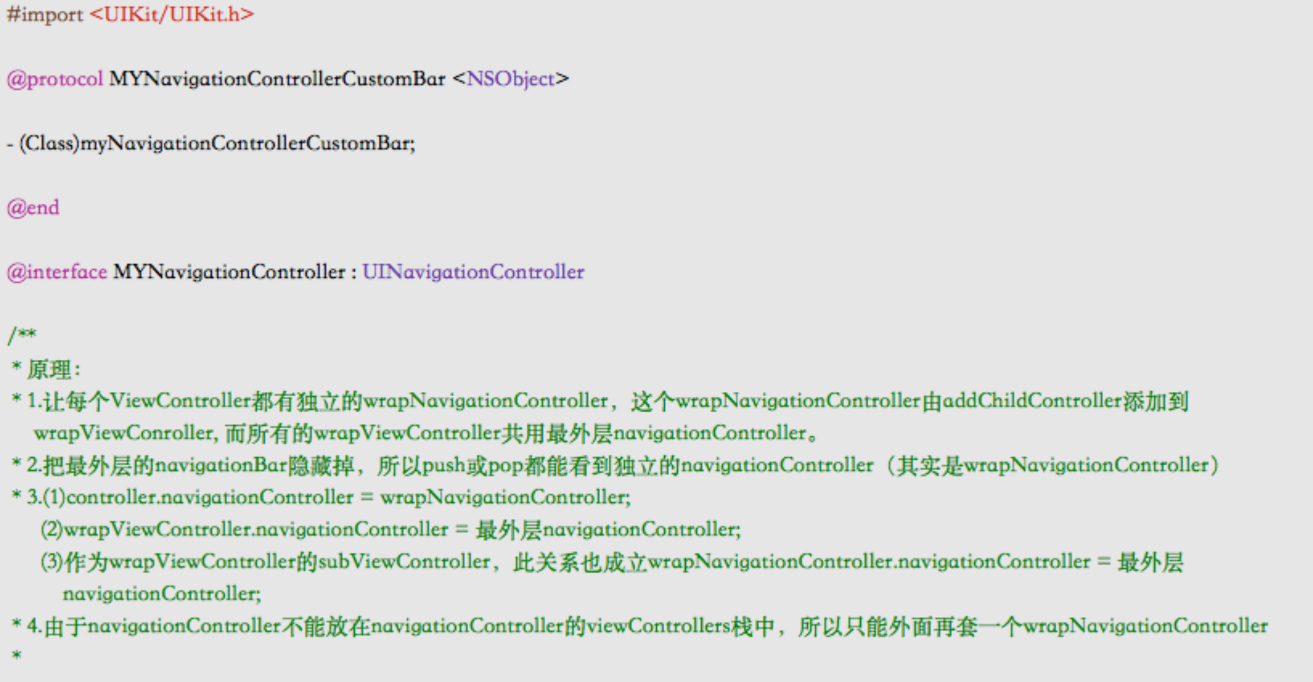
h MYWrapViewController.h

m MYWrapViewController.m

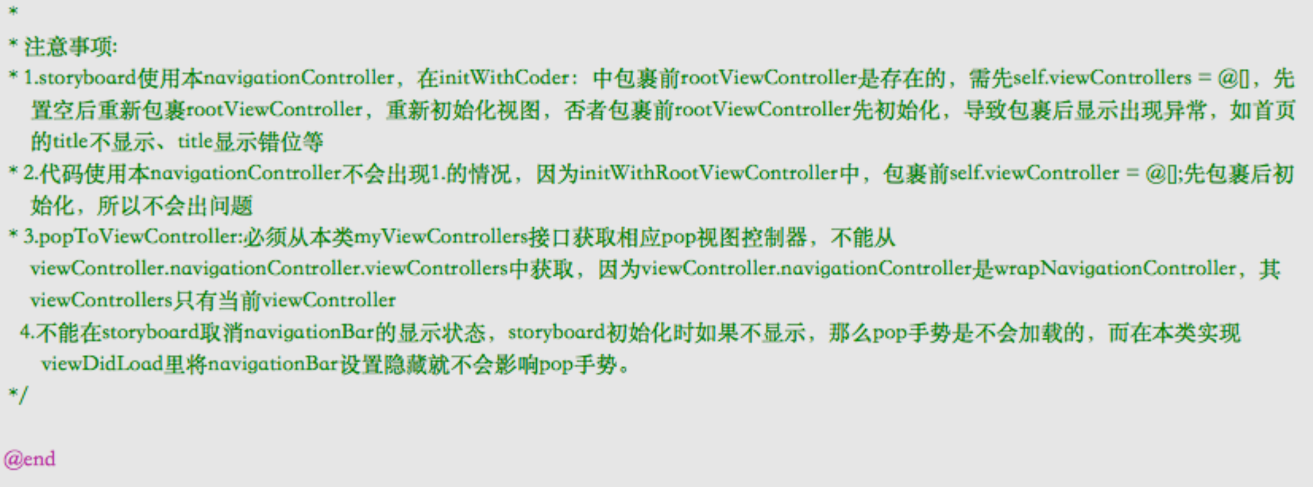
h UIViewController+MYWrapViewController.h

m UIViewController+MYWrapViewController.m

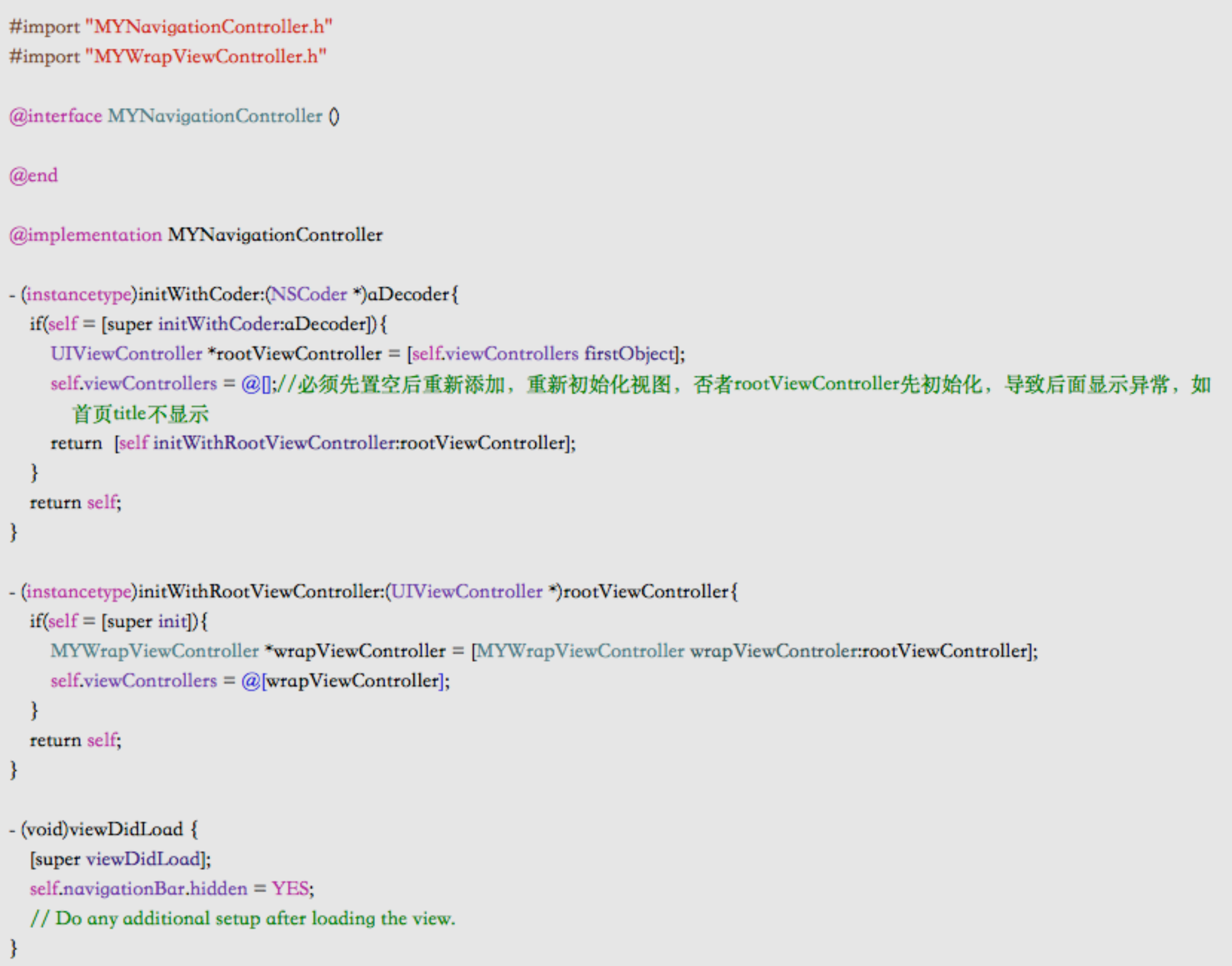
(2) 具体封装实现



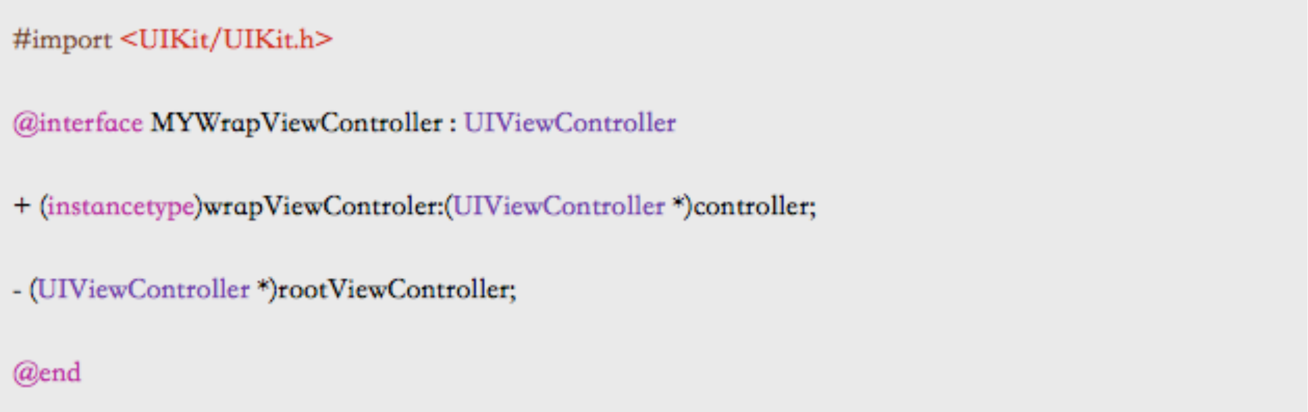
```
#import <UIKit/UIKit.h>

@protocol MYNavigationControllerCustomBar <NSObject>

- (Class)myNavigationControllerCustomBar;

@end

@interface MYNavigationController : UINavigationController

/**
 * 原理:
 * 1.让每个ViewController都有独立的wrapNavigationController，这个wrapNavigationController由addChildController添加到
   wrapViewConroller, 而所有的wrapViewController共用最外层navigationController。
 * 2.把最外层的navigationBar隐藏掉，所以push或pop都能看到独立的navigationController（其实是wrapNavigationController）
 * 3.(1)controller.navigationController = wrapNavigationController;
    (2)wrapViewController.navigationController = 最外层navigationController;
    (3)作为wrapViewController的subViewController，此关系也成立wrapNavigationController.navigationController = 最外层
       navigationController;
 * 4.由于navigationController不能放在navigationController的viewControllers栈中，所以只能外面再套一个wrapNavigationController
 *
```



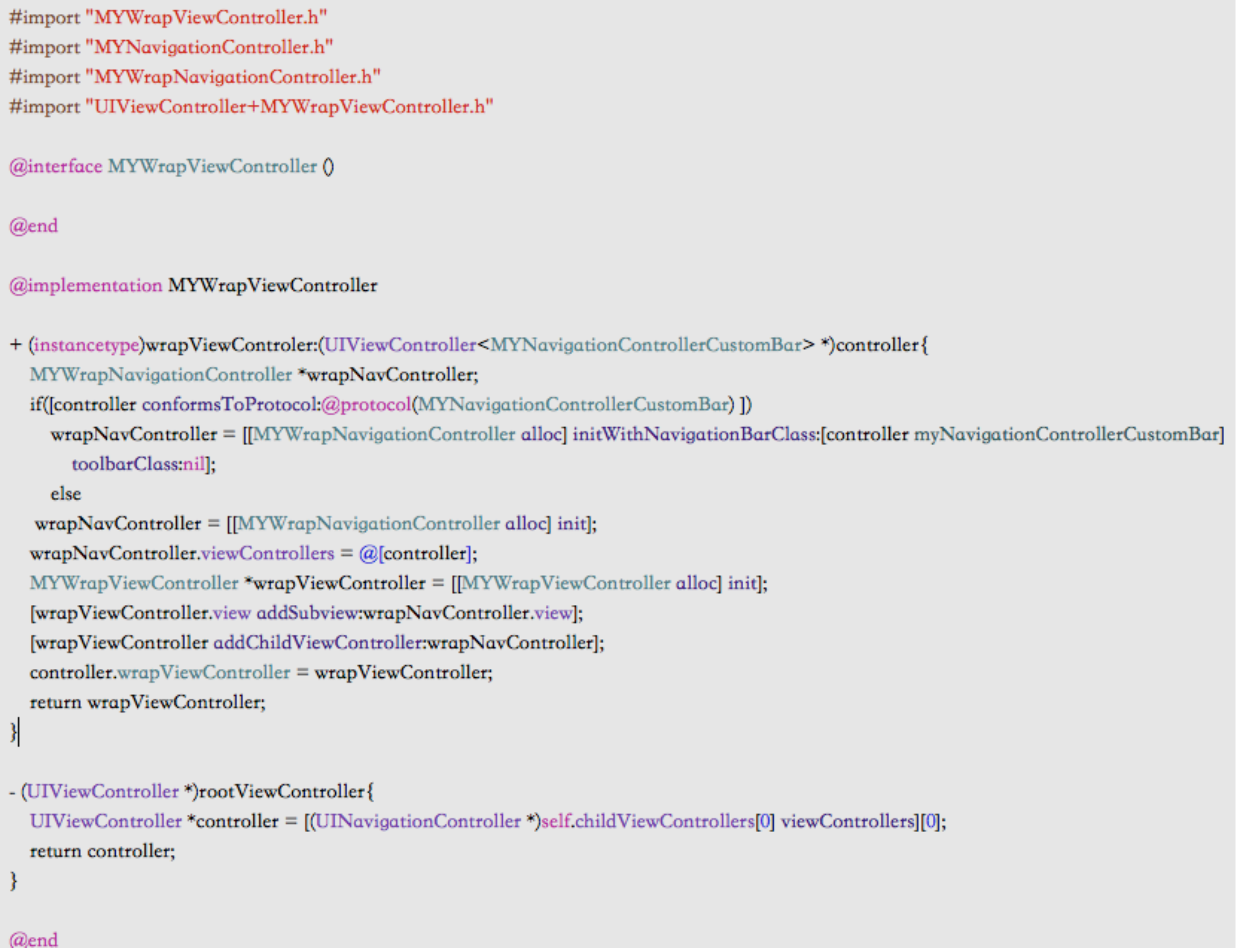
```
 *
 * 注意事项:
 * 1.storyboard使用本navigationController，在initWithCoder：中包裹前rootViewController是存在的，需先self.viewControllers = @[]，先
   置空后重新包裹rootViewController，重新初始化视图，否者包裹前rootViewController先初始化，导致包裹后显示出现异常，如首页
   的title不显示、title显示错位等
 * 2.代码使用本navigationController不会出现1.的情况，因为initWithRootViewController中，包裹前self.viewController = @[];先包裹后初
   始化，所以不会出问题
 * 3.popToViewController:必须从本类myViewControllers接口获取相应pop视图控制器，不能从
   viewController.navigationController.viewControllers中获取，因为viewController.navigationController是wrapNavigationController，其
   viewControllers只有当前viewController
  4.不能在storyboard取消navigationBar的显示状态，storyboard初始化时如果不显示，那么pop手势是不会加载的，而在本类实现
    viewDidLoad里将navigationBar设置隐藏就不会影响pop手势。
*/

@end
```



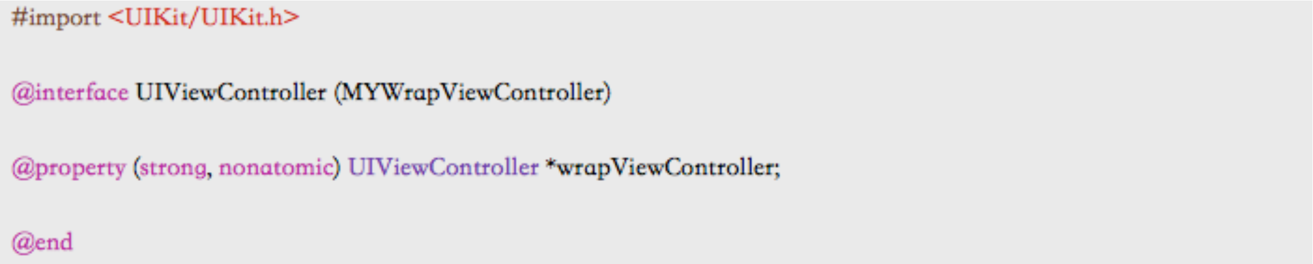
```
#import "MYNavigationController.h"
#import "MYWrapViewController.h"

@interface MYNavigationController ()

@end

@implementation MYNavigationController

- (instancetype)initWithCoder:(NSCoder *)aDecoder{
   if(self = [super initWithCoder:aDecoder]){
      UIViewController *rootViewController = [self.viewControllers firstObject];
      self.viewControllers = @[];//必须先置空后重新添加，重新初始化视图，否者rootViewController先初始化，导致后面显示异常，如
        首页title不显示
      return  [self initWithRootViewController:rootViewController];
   }
   return self;
}

- (instancetype)initWithRootViewController:(UIViewController *)rootViewController{
   if(self = [super init]){
      MYWrapViewController *wrapViewController = [MYWrapViewController wrapViewControler:rootViewController];
      self.viewControllers = @[wrapViewController];
   }
   return self;
}

- (void)viewDidLoad {
   [super viewDidLoad];
   self.navigationBar.hidden = YES;
   // Do any additional setup after loading the view.
}
```



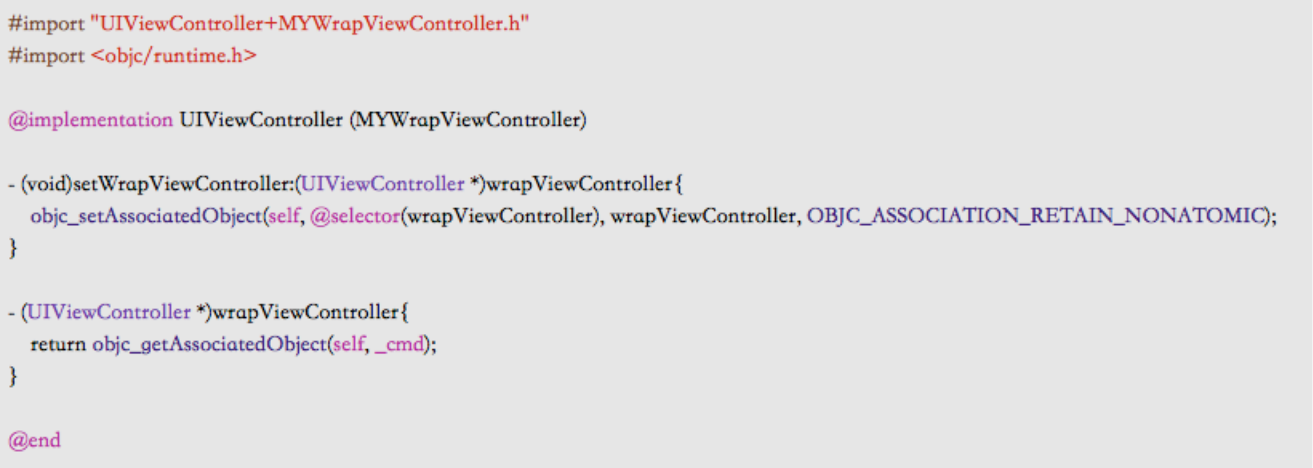
```
#import <UIKit/UIKit.h>

@interface MYWrapViewController : UIViewController

+ (instancetype)wrapViewControler:(UIViewController *)controller;

- (UIViewController *)rootViewController;

@end
```



```
#import "MYWrapViewController.h"
#import "MYNavigationController.h"
#import "MYWrapNavigationController.h"
#import "UIViewController+MYWrapViewController.h"

@interface MYWrapViewController ()

@end

@implementation MYWrapViewController

+ (instancetype)wrapViewControler:(UIViewController<MYNavigationControllerCustomBar> *)controller{
    MYWrapNavigationController *wrapNavController;
    if([controller conformsToProtocol:@protocol(MYNavigationControllerCustomBar) ])
        wrapNavController = [[MYWrapNavigationController alloc] initWithNavigationBarClass:[controller myNavigationControllerCustomBar]
            toolbarClass:nil];
        else
     wrapNavController = [[MYWrapNavigationController alloc] init];
    wrapNavController.viewControllers = @[controller];
    MYWrapViewController *wrapViewController = [[MYWrapViewController alloc] init];
    [wrapViewController.view addSubview:wrapNavController.view];
    [wrapViewController addChildViewController:wrapNavController];
    controller.wrapViewController = wrapViewController;
    return wrapViewController;
}

- (UIViewController *)rootViewController{
    UIViewController *controller = [(UINavigationController *)self.childViewControllers[0] viewControllers][0];
    return controller;
}

@end
```



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

@interface UIViewController (MYWrapViewController)

@property (strong, nonatomic) UIViewController *wrapViewController;

@end
```



```
#import "UIViewController+MYWrapViewController.h"
#import <objc/runtime.h>

@implementation UIViewController (MYWrapViewController)

- (void)setWrapViewController:(UIViewController *)wrapViewController{
    objc_setAssociatedObject(self, @selector(wrapViewController), wrapViewController, OBJC_ASSOCIATION_RETAIN_NONATOMIC);
}

- (UIViewController *)wrapViewController{
    return objc_getAssociatedObject(self, _cmd);
}

@end
```