MYNavigationViewControllerV2 封装

1. 简介：看这个标题，大家可能会奇怪，怎么又是 NavigationController 的封装，之前就有写过两次了，分别在文档“MYNavigationController 封装”和“UIViewControllerTransitioning”中。每次封装都有个封装需求或者说封装的意义。那接下来我们就来探讨下，此次的封装，是否有意义。

2. 封装意图：

首先，解释下，我们封装的是叫 MYNavigationViewControllerV2，之所以有个 V2，一点是之前已经有过一次封装了；还有就是，我认为，此次的封装是一年多积累来对导航封装思想的一个升级把。为什么这么说呢？我们一步步来解惑。

(1)有一点，先明确的是自定义导航的目的：解决系统共用导航存在的问题，让每个 controller 从视觉上或者从本质上拥有一个独立的导航栏。

1)什么是视觉上拥有呢？那就是我 UIViewControllerTransitioning 做的封装，主要是利用转场动画来实现，push 动画执行时，将当前页面导航栏截屏下来 add 到 fromVC 上，然后因为真正的导航栏在 toVC 上，那么，就可以通过转场动画中对 fromVC、真导航栏、toVC 三者的移动来完成动画，结束后将 imageview 移除；Pop 动画执行时，将当前页面导航栏截屏下来 add 到 fromVC，并获取 push 时截屏的图片栈顶取 toVC 的导航栏图片 add 到 ToVC，并把真导航栏移到最底层(看不见)，结束后将 image 移除，同时 push 图片栈栈顶图出栈，并把真导航栏移到最顶层，完成 pop 动画。

2)什么是本质上拥有呢？相比 1)中介绍的，所有页面还是共用了导航栏，只是通过动画给予用户感觉独立而已。相反的，MYNavigationViewController(V2)这两次封装结果，都是每个页面都有独立的导航栏。先介绍下之前封装 MYNavigationViewController 的原理：每个页面结构都是套了一个 navigationController，然后再套一个 wrapViewController，push 的时候要对 viewController 进行包裹得到 wrapViewController，然后进行 push，然后把最顶层导航栏隐藏，也就实现只有每个独立的 wrapViewController 的导航栏可见。当然 push 时候再最外层导航栏 push 方法里对待 push 的页面的导航栏进行设置返回按钮啊啥的，还有添加新页面导航栏返回事件等；这个封装理解起来有点绕啊，这里再讲下这种封装后的页面结构：

A push B：实际是 A.navigationController.navigationController，也就是 push 真正的执行者是外层的 navigationController，因为 wrapViewController addCildViewController:wrapNavigationController，所以 A.navigationController，也就是 wrapNavigationController 的 navigationController 和 wrapViewController 的 navigationController 是一样的

3)接下来，抛砖引玉，即将进入正题了：此次封装的 V2 版本有什么意义呢？先分析下前面两者的弊端，1)中介绍的封装只是视觉上实现的，本质还是共用导航栏，这会引来很多不必要的设置操作；我们更倾向于每个页面不用什么设置，默认就有独

立的导航栏； 2)中介绍的封装虽然实现了我们想要的效果了，但是，显而易见，代价太大了，原本 A push B ，本来栈里就两个 controller，现在栈里有两个 controller，每个 controller 里有一个导航栏，导航栏里还有一个最终的 A 或 B controller。所以，我在想，有什么办法能够低消耗地，非常简单的让每个页面，从本质上拥有一个独立的导航栏呢？

4)没错，这就是 V2 进行封装的意义所在了，V2 里，我们采用自定义导航栏(普通 UIView),通过封装导航栏控制器，再 push 方法里，将 view 初始化，并 add 到 toVC，并且设置导航按钮事件代理为 toVC，这种方式，很简单的让每个页面有了独立的导航栏，还有，另外一点很重要的是：V2 借鉴了 1)中自定义转场动画的实现，自定义了 push pop 动画，新增全屏 pop 手势和边缘手势以及对他们的管理，解决了系统边缘手势强制转换成 pan 手势添加到 controller 的 view 或者 navigationController 的 view 后，页面容易造成页面卡死(同时其他导航栈页面正常)的奇怪现象(应该是频繁调用系统边缘手势触发方法导致了 navigationController 功能错乱)。因为项目中确实出现了这个问题，而且非常频繁，所以才有了这次 V2 封装的意图。

3. MYNavigationViewControllerV2 的封装过程：

由于时间有限，先想代码实现给出，后续空时再进行详细讲解。内部有关于转场动画的知识，所以看之前希望大家先把之前的“UIViewControllerTransitioning”篇看完，因为里面有对转场动画进行详细的介绍，本次关于转场动画仅做了一些修改。

(1)　自定义导航栏 NavigationView 的封装(在同事封装的基础上进行优化的，哈哈)

```
//  NavigationView.h
//  GuDaShi
//
//  Created by songzhaojie on 17/1/12.
//  Copyright © 2017年 songzhaojie. All rights reserved.
//
#import "MYImageButton.h"
#import <UIKit/UIKit.h>

@protocol  NavigationViewViewDlegate<NSObject>
@optional
-(void)navigationViewLeftDlegate;
-(void)navigationViewReghtDlegate;
@end

@interface NavigationView : UIView
@property(nonatomic,strong)MYImageButton *leftBtn;
@property(nonatomic,strong) UILabel *zHongJianlable;
@property(nonatomic,strong)MYImageButton* reghtBtn;

@property(nonatomic,weak)id <NavigationViewViewDlegate>  delegate;
//左边设置图片
- (void)leftBtnSheZhiImage:(NSString *)image withText:(NSString *)text  withHidden:(BOOL)hidded;//autolayout
-(void)leftBtnSheZhiImage:(NSString *)image withText:(NSString *)text withHidden:(BOOL)hidded withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height;

//中间设置lable
-(void)zhongJianLableSheZhiLable:(NSString *)lable withZiShiYing:(BOOL)ZiShiYing  withFont:(float)font;//autolayout
-(void)zhongJianLableSheZhiLable:(NSString *)lable withZiShiYing:(BOOL)ZiShiYing  withFont:(float)font withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height;

//右边设置图片
-(void)reghtBtnSheZhiImage:(NSString *)image withText:(NSString *)text withHidden:(BOOL)hidded;//autolayout
-(void)reghtBtnSheZhiImage:(NSString *)image withText:(NSString *)text withHidden:(BOOL)hidded withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height withImageWidth:(float)imageWidth withImageHeight:(float)imageHeig
@end
```

里面有用到 MYImageButton，因为我需要指定导航栏按钮中图片的位置和大小通过读取 image 的宽高结合 MYImageButton 的设置来实现大按钮小图片。这个按钮的封装也很简单，继承 UIButton 后重写系统方法即可

```
-(CGRect)imageRectForContentRect:(CGRect)contentRect{
    if(_imageFrame.size.width > 0){
        return _imageFrame;
    }
    return [super imageRectForContentRect:contentRect];
}

-(CGRect)titleRectForContentRect:(CGRect)contentRect
{
    if(_labelFrame.size.width > 0){
        return self.labelFrame;
    }
    return [super titleRectForContentRect:contentRect];
}
```

```
#import "NavigationView.h"
#import "UIView+CWNView.h"

@implementation NavigationView

/*
 // Only override drawRect: if you perform custom drawing.
 // An empty implementation adversely affects performance during animation.
 - (void)drawRect:(CGRect)rect {
 // Drawing code
 }
 */
-(id)initWithFrame:(CGRect)frame
{
    self=[super initWithFrame:frame];
    if (self){
    }

    return self;
}
```

```
- (void)leftBtnSheZhiImage:(NSString *)image withText:(NSString *)text  withHidden:(BOOL)hidded{
    if(!_leftBtn){
        _leftBtn=[MYImageButton buttonWithType:UIButtonTypeCustom];
        [_leftBtn setTitleColor:[UIColor whiteColor] forState:UIControlStateNormal];
        _leftBtn.titleLabel.font=[UIFont systemFontOfSize:15];
        [_leftBtn addTarget:self action:@selector(Leftbtn) forControlEvents:UIControlEventTouchUpInside];
        [self addSubview:_leftBtn];

        [_leftBtn cwn_makeConstraints:^(UIView *maker) {
            maker.leftToSuper(2.5).topToSuper(20).bottomToSuper(0).width(100);
        }];
    }

//    if([image length]){//图片
        UIImage *imageView = [UIImage imageNamed:image];
        _leftBtn.imageFrame=CGRectMake(10, (self.frame.size.height - 20) / 2 - imageView.size.height / 2, imageView.size.width, imageView.size.height);
        [_leftBtn setImage:imageView forState:UIControlStateNormal];
//    }

//    if([text length]){//文字
        _leftBtn.labelFrame = CGRectMake(10, 0, 90, self.frame.size.height - 20);
        [_leftBtn setTitle:text forState:UIControlStateNormal];
        [_leftBtn.titleLabel setTextAlignment:NSTextAlignmentLeft];
//    }

    _leftBtn.hidden=hidded;
}
-(void)leftBtnSheZhiImage:(NSString *)image  withText:(NSString *)text withHidden:(BOOL)hidded withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height
{
    [self leftBtnSheZhiImage:image withText:text withHidden:hidded];
}

- (void)zhongJianLableSheZhiLable:(NSString *)lable withZiShiYing:(BOOL)ZiShiYing withFont:(float)font{
    if(!_zHongJianlable){
        _zHongJianlable=[[UILabel alloc]init];
        [self addSubview:_zHongJianlable];

        [_zHongJianlable cwn_makeConstraints:^(UIView *maker) {
            maker.centerXtoSuper(0).topToSuper(20).bottomToSuper(0);
        }];

        _zHongJianlable.adjustsFontSizeToFitWidth=ZiShiYing;
        _zHongJianlable.textAlignment=NSTextAlignmentCenter;
        _zHongJianlable.font=[UIFont systemFontOfSize:font];
        _zHongJianlable.textColor=[UIColor whiteColor];
    }
    _zHongJianlable.text=lable;
}
-(void)zhongJianLableSheZhiLable:(NSString *)lable withZiShiYing:(BOOL)ZiShiYing  withFont:(float)font withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height
{
    [self zhongJianLableSheZhiLable:lable withZiShiYing:ZiShiYing withFont:font];
}


- (void)reghtBtnSheZhiImage:(NSString *)image withText:(NSString *)text withHidden:(BOOL)hidded{
    if(!_reghtBtn){
        _reghtBtn=[MYImageButton buttonWithType:UIButtonTypeCustom];
        [_reghtBtn addTarget:self action:@selector(reghtBtn1) forControlEvents:UIControlEventTouchUpInside];
        [_reghtBtn setTitleColor:[UIColor whiteColor] forState:UIControlStateNormal];
        _reghtBtn.titleLabel.font=[UIFont systemFontOfSize:15];
        _reghtBtn.titleLabel.textColor=[UIColor whiteColor];
        [self addSubview:_reghtBtn];

        [_reghtBtn cwn_makeConstraints:^(UIView *maker) {
            maker.rightToSuper(2.5).topToSuper(20).bottomToSuper(0).width(100);
        }];
    }

//    if([image length]){//图片
        UIImage *imageView = [UIImage imageNamed:image];
        _reghtBtn.imageFrame=CGRectMake(100 - imageView.size.width - 10, (self.frame.size.height - 20) / 2 - imageView.size.height / 2, imageView.size.width, imageView.size.height);
        [_reghtBtn setImage:[UIImage imageNamed:image] forState:UIControlStateNormal];
//    }

//    if([text length]){//文字
        _reghtBtn.labelFrame = CGRectMake(0, 0, 90, self.frame.size.height - 20);
         [_reghtBtn setTitle:text forState:UIControlStateNormal];
        [_reghtBtn.titleLabel setTextAlignment:NSTextAlignmentRight];
//    }


    _reghtBtn.hidden=hidded;
}
-(void)reghtBtnSheZhiImage:(NSString *)image withText:(NSString *)text withHidden:(BOOL)hidded withfloatX:(float)floatX withfloatY:(float)floatY withWidth:(float)width withHeight:(float)height withImageWidth:(float)imageWidth withImageHeight:(float)imageHeight
{
    [self reghtBtnSheZhiImage:image withText:text withHidden:hidded];
}



-(void)Leftbtn{
    if ([_delegate respondsToSelector:@selector(navigationViewLeftDlegate)]) {
        [_delegate navigationViewLeftDlegate];
    }
}
-(void)reghtBtn1{
    if ([_delegate respondsToSelector:@selector(navigationViewReghtDlegate)]) {
        [_delegate navigationViewReghtDlegate];
    }
}
```

(2) MYNavigationControllerV2 头文件：

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

@interface UIViewController (MYNav)<NavigationViewViewDlegate>

@property (strong, nonatomic) NavigationView *navigationBar;

@property (strong, nonatomic) NSString *navigationBar_title;
@property (strong, nonatomic) NSString *navigationBar_leftImage;
@property (strong, nonatomic) NSString *navigationBar_rightImage;
@property (strong, nonatomic) NSString *navigationBar_leftTitle;
@property (strong, nonatomic) NSString *navigationBar_rightTitle;
@property (weak, nonatomic) id <NavigationViewViewDlegate> navigationBar_delegate;
@property (assign, nonatomic) BOOL navigationBar_hidden;

@end
```

```
@interface UINavigationController (MYNav)

/**
 * 是否需要全屏返回手势
 *
 * @note  默认为NO
 * @note  设置前提：使用MYCustomNavigationController导航，否则设置无效，始终为NO
 */
@property (assign, nonatomic) BOOL popPanGestureEnabled;

/**
 * 是否需要边缘返回手势
 *
 * @note  默认为NO
 * @note  设置前提：使用MYCustomNavigationController导航，否则设置无效，始终为NO
 */
@property (assign, nonatomic) BOOL popEdgePanGestureEnabled;

@end
```

```
@interface MYNavigationControllerV2 : UINavigationController

@property (assign, nonatomic) BOOL interactPopPanGestureEnabled;//同MYCustomNav分类中的popPanGestureEnabled
@property (assign, nonatomic) BOOL interactPopEdgePanGestureEnabled;//同MYCustomNav分类中的popEdgePanGestureEnabled

@end
```

(3) MYNavigationControllerV2 实现文件：

```
#import "MYNavigationControllerV2.h"
#import "MYNavControllerTransitioningDelegateV2.h"
#import <objc/runtime.h>

@implementation UIViewController (MYNav)

- (NavigationView *)navigationBar{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    NavigationView *navigationBar = objc_getAssociatedObject(self, _cmd);
    return navigationBar;
}
- (void)setNavigationBar:(NavigationView *)navigationBar{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    objc_setAssociatedObject(self, @selector(navigationBar), navigationBar, OBJC_ASSOCIATION_RETAIN_NONATOMIC);
}

- (NSString *)navigationBar_title{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return objc_getAssociatedObject(self, _cmd);
}
- (void)setNavigationBar_title:(NSString *)navigationBar_title{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] zhongJianLableSheZhiLable:navigationBar_title withZiShiYing:NO withFont:17];
    objc_setAssociatedObject(self, @selector(navigationBar_title), navigationBar_title, OBJC_ASSOCIATION_COPY);
}
```

```
- (NSString *)navigationBar_leftImage{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return objc_getAssociatedObject(self, _cmd);
}
- (void)setNavigationBar_leftImage:(NSString *)navigationBar_leftImage{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] leftBtnSheZhiImage:navigationBar_leftImage withText:navigationBar_leftImage withHidden:NO];
    objc_setAssociatedObject(self, @selector(navigationBar_leftImage), navigationBar_leftImage, OBJC_ASSOCIATION_COPY);
}

- (NSString *)navigationBar_rightImage{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return objc_getAssociatedObject(self, _cmd);
}
- (void)setNavigationBar_rightImage:(NSString *)navigationBar_rightImage{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] reghtBtnSheZhiImage:navigationBar_rightImage withText:@"" withHidden:NO];
    objc_setAssociatedObject(self, @selector(navigationBar_rightImage), navigationBar_rightImage, OBJC_ASSOCIATION_COPY);
}

- (NSString *)navigationBar_leftTitle{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return objc_getAssociatedObject(self, _cmd);
}
- (void)setNavigationBar_leftTitle:(NSString *)navigationBar_leftTitle{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] leftBtnSheZhiImage:@"" withText:navigationBar_leftTitle withHidden:NO];
    objc_setAssociatedObject(self, @selector(navigationBar_leftTitle), navigationBar_leftTitle, OBJC_ASSOCIATION_COPY);
}



- (NSString *)navigationBar_rightTitle{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return objc_getAssociatedObject(self, _cmd);
}
- (void)setNavigationBar_rightTitle:(NSString *)navigationBar_rightTitle{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] reghtBtnSheZhiImage:@"" withText:navigationBar_rightTitle withHidden:NO];
    objc_setAssociatedObject(self, @selector(navigationBar_rightTitle), navigationBar_rightTitle, OBJC_ASSOCIATION_COPY);
}

- (id<NavigationViewViewDlegate>)navigationBar_delegate{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return nil;
    return [self navigationBar].delegate;
}
- (void)setNavigationBar_delegate:(id<NavigationViewViewDlegate>)navigationBar_delegate{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    [[self navigationBar] setDelegate:navigationBar_delegate];
}

- (BOOL)navigationBar_hidden{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return NO;
    return [objc_getAssociatedObject(self, _cmd) boolValue];
}
- (void)setNavigationBar_hidden:(BOOL)navigationBar_hidden{
    if([self isMemberOfClass:[MYNavigationControllerV2 class]])
        return;
    objc_setAssociatedObject(self, @selector(navigationBar_hidden), [NSNumber numberWithBool:navigationBar_hidden], OBJC_ASSOCIATION_RETAIN_NONATOMIC);
    [[self navigationBar] setHidden:navigationBar_hidden];
}

@end
```

```
@implementation UINavigationController (MYNav)

- (void)setPopPanGestureEnabled:(BOOL)popGestureEnabled{
   if([self isKindOfClass:[MYNavigationControllerV2 class]]){
      [(MYNavigationControllerV2 *)self setInteractPopPanGestureEnabled:popGestureEnabled];
   }
}
- (BOOL)popPanGestureEnabled{
   if([self isKindOfClass:[MYNavigationControllerV2 class]]){
     return  [(MYNavigationControllerV2 *)self interactPopPanGestureEnabled];
   }
   return NO;
}

- (void)setPopEdgePanGestureEnabled:(BOOL)popEdgePanGestureEnabled{
   if([self isKindOfClass:[MYNavigationControllerV2 class]]){
      [(MYNavigationControllerV2 *)self setInteractPopEdgePanGestureEnabled:popEdgePanGestureEnabled];
   }
}
- (BOOL)popEdgePanGestureEnabled{
   if([self isKindOfClass:[MYNavigationControllerV2 class]]){
      return  [(MYNavigationControllerV2 *)self interactPopEdgePanGestureEnabled];
   }
   return NO;
}

@end


@interface MYNavigationControllerV2 ()

@property (nonatomic, strong) MYNavControllerTransitioningDelegateV2 *navDelegate;

@end

@implementation MYNavigationControllerV2

- (void)viewDidLoad {
   [super viewDidLoad];
   self.navDelegate = [[MYNavControllerTransitioningDelegateV2 alloc] initWithNavigationController:self];
   self.navDelegate.interactivePopPanGestureRecognizer.enabled = NO;
   self.navDelegate.interactivePopEdgePanGestureRecognizer.enabled = NO;
   self.navigationBar.hidden = YES;

   UIViewController *rootViewController = [self.viewControllers firstObject];
   NavigationView *navigationBar = [[NavigationView alloc]initWithFrame:CGRectMake(0, 0,[UIScreen mainScreen].bounds.size.width, 64)];
   navigationBar.backgroundColor=[UIColor colorWithRed:201/255.0 green:8/255.0 blue:19/255.0 alpha:1];
   [navigationBar leftBtnSheZhiImage:@"" withText:@"" withHidden:NO];
   [navigationBar zhongJianLableSheZhiLable:@"" withZiShiYing:NO withFont:17];
   [navigationBar reghtBtnSheZhiImage:@"" withText:@"" withHidden:NO];
   [rootViewController setNavigationBar:navigationBar];
   [rootViewController.view addSubview:navigationBar];
   [rootViewController setNavigationBar_delegate:rootViewController];
}

- (void)viewDidAppear:(BOOL)animated{
   [super viewDidAppear:animated];
}

- (void)didReceiveMemoryWarning {
   [super didReceiveMemoryWarning];
   // Dispose of any resources that can be recreated.
}

#pragma mark 重写父类方法

- (void)pushViewController:(UIViewController *)viewController animated:(BOOL)animated{
   NavigationView *navigationBar = [[NavigationView alloc]initWithFrame:CGRectMake(0, 0,[UIScreen mainScreen].bounds.size.width, 64)];
   navigationBar.backgroundColor=[UIColor colorWithRed:201/255.0 green:8/255.0 blue:19/255.0 alpha:1];
   [navigationBar leftBtnSheZhiImage:@"zhiboLeft" withText:@"" withHidden:NO];
   [navigationBar zhongJianLableSheZhiLable:@"" withZiShiYing:NO withFont:17];
   [navigationBar reghtBtnSheZhiImage:@"" withText:@"" withHidden:NO];
   [viewController setNavigationBar:navigationBar];
   [viewController.view addSubview:navigationBar];
   [viewController setNavigationBar_delegate:viewController];

   [super pushViewController:viewController animated:animated];
}

- (UIViewController *)popViewControllerAnimated:(BOOL)animated{
  return [super popViewControllerAnimated:animated];
}

- (NSArray<UIViewController *> *)popToViewController:(UIViewController *)viewController animated:(BOOL)animated{
   return [super popToViewController:viewController animated:animated];
}

- (NSArray<UIViewController *> *)popToRootViewControllerAnimated:(BOOL)animated{
   return [super popToRootViewControllerAnimated:animated];
}
```

```
#pragma mark set方法

- (void)setInteractPopPanGestureEnabled:(BOOL)interactPopGestureEnabled{
    self.navDelegate.interactivePopPanGestureRecognizer.enabled = interactPopGestureEnabled;
    if(interactPopGestureEnabled == YES){//与边缘、系统边缘手势互斥
        self.navDelegate.interactivePopEdgePanGestureRecognizer.enabled = !interactPopGestureEnabled;
        self.interactivePopGestureRecognizer.enabled = !interactPopGestureEnabled;
    }else{//同步禁用边缘、系统边缘手势
        self.navDelegate.interactivePopEdgePanGestureRecognizer.enabled = interactPopGestureEnabled;
        self.interactivePopGestureRecognizer.enabled = interactPopGestureEnabled;
    }

    _interactPopPanGestureEnabled = interactPopGestureEnabled;
}

- (void)setInteractPopEdgePanGestureEnabled:(BOOL)interactPopEdgePanGestureEnabled{
    self.navDelegate.interactivePopEdgePanGestureRecognizer.enabled = interactPopEdgePanGestureEnabled;
    if(interactPopEdgePanGestureEnabled == YES){//与全屏、系统边缘手势互斥
        self.navDelegate.interactivePopPanGestureRecognizer.enabled = !interactPopEdgePanGestureEnabled;
        self.interactivePopGestureRecognizer.enabled = !interactPopEdgePanGestureEnabled;
    }else{//同步禁用全屏、系统边缘手势
        self.navDelegate.interactivePopEdgePanGestureRecognizer.enabled = interactPopEdgePanGestureEnabled;
        self.interactivePopGestureRecognizer.enabled = interactPopEdgePanGestureEnabled;
    }

    _interactPopEdgePanGestureEnabled = interactPopEdgePanGestureEnabled;
}

@end
```

(4) MYNavControllerTransitioningDelegateV2 头文件

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

/*
 本类定义了MYNavigationController的代理，主要处理push、pop动画触发事件
 */

@interface MYNavControllerTransitioningDelegateV2 : NSObject<UINavigationControllerDelegate>

@property (strong, nonatomic) UIPanGestureRecognizer *interactivePopPanGestureRecognizer;//全屏返回手势
@property (strong, nonatomic) UIScreenEdgePanGestureRecognizer *interactivePopEdgePanGestureRecognizer;//边缘返回手势

- (instancetype)initWithNavigationController:(UINavigationController *)controller;

@end
```

(5) MYNavControllerTransitioningDelegateV2 实现文件

```
#import "MYNavControllerTransitioningDelegateV2.h"
#import "MYViewControllerAnimatedTransitioningV2.h"

@interface MYNavControllerTransitioningDelegateV2 ()

@property (assign, nonatomic) BOOL interActiving;

@property (strong, nonatomic) UIPercentDrivenInteractiveTransition* percentDrivenInteractiveTransition;
@property (strong, nonatomic) MYViewControllerAnimatedTransitioningV2 *animation;

@property (strong, nonatomic) UINavigationController *navController;

@end

@implementation MYNavControllerTransitioningDelegateV2

- (instancetype)initWithNavigationController:(UINavigationController *)controller{
    if(self = [super init]){
        _navController = controller;
        [controller setDelegate:self];

        _animation = [[MYViewControllerAnimatedTransitioningV2 alloc] init];
        _percentDrivenInteractiveTransition = [[UIPercentDrivenInteractiveTransition alloc] init];

        _interactivePopPanGestureRecognizer = [[UIPanGestureRecognizer alloc] initWithTarget:self action:@selector(handleGesture:)];
        [_navController.view addGestureRecognizer:_interactivePopPanGestureRecognizer];

        _interactivePopEdgePanGestureRecognizer = [[UIScreenEdgePanGestureRecognizer alloc] initWithTarget:self action:@selector(handleGesture:)];
        _interactivePopEdgePanGestureRecognizer.edges = UIRectEdgeLeft;
        [_navController.view addGestureRecognizer:_interactivePopEdgePanGestureRecognizer];
    }
    return self;
}
```

```
#pragma mark private methods

-(void)handleGesture:(UIPanGestureRecognizer *)gesture {
    UIView* view = self.navController.view;
    CGPoint translation = [gesture translationInView:gesture.view];

    switch (gesture.state) {
        case UIGestureRecognizerStateBegan: {
            _interActiving = YES;
            if([self.navController.viewControllers count] > 1){
                [self.navController popViewControllerAnimated:YES];
            }
            break;
        }
        case UIGestureRecognizerStateChanged: {
            CGFloat fraction = translation.x / view.bounds.size.width;
            if(fraction < 0)
                fraction = 0;
                [_percentDrivenInteractiveTransition updateInteractiveTransition:fraction];
            break;
        }
        case UIGestureRecognizerStateCancelled:
        case UIGestureRecognizerStateEnded: {
            _interActiving = NO;
            CGFloat fraction = translation.x / view.bounds.size.width;
            if ((fraction > 0 && fraction < 0.5) || gesture.state == UIGestureRecognizerStateCancelled || fraction <=0) {
                [_percentDrivenInteractiveTransition cancelInteractiveTransition];
            } else {
                [_percentDrivenInteractiveTransition finishInteractiveTransition];
            }
            break;
        }
        default:
            break;
    }
}
```

```
#pragma mark UINavigationControllerDelegate

- (id <UIViewControllerAnimatedTransitioning>)navigationController:(UINavigationController *)navigationController animationControllerForOperation:(UINavigationControllerOperation)operation fromViewController:(UIViewController *)fromVC toViewController:
   (UIViewController *)toVC {
   if (operation == UINavigationControllerOperationPop) {
      _animation.type  = UIViewControllerPresentTransitionTypePop;
      return _animation;
   }else  if (operation == UINavigationControllerOperationPush) {
      _animation.type  = UIViewControllerPresentTransitionTypePush;
      return _animation;
   }
   return nil;
}

- (id<UIViewControllerInteractiveTransitioning>)navigationController:(UINavigationController *)navigationController interactionControllerForAnimationController:(id<UIViewControllerAnimatedTransitioning>)animationController{
   return self.interActiving ? self.percentDrivenInteractiveTransition : nil;
}
```

(6) MYViewControllerAnimatedTransitioningV2 头文件

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

/*
 本类仅定义了固定动画(仿系统默认动画)，可自由发挥，定义各种动画，以枚举区分即可
 */

typedef NS_ENUM(NSUInteger, UIViewControllerPresentTransitionType) {
    UIViewControllerPresentTransitionTypePresent = 0,//present动画
    UIViewControllerPresentTransitionTypeDismiss,//dismiss动画
    UIViewControllerPresentTransitionTypePush,//push动画
    UIViewControllerPresentTransitionTypePop//pop动画
};

@interface MYViewControllerAnimatedTransitioningV2 : NSObject<UIViewControllerAnimatedTransitioning>

@property (assign, nonatomic) UIViewControllerPresentTransitionType type;

@end
```

(7) MYViewControllerAnimatedTransitioningV2 实现文件

```objc
#import "MYViewControllerAnimatedTransitioningV2.h"
#import "UIView+FrameProperty.h"

static const float kTransitionDuration = .28;

@interface MYViewControllerAnimatedTransitioningV2 ()

@property (strong, nonatomic) UIView *tabBarSuperview;

@end

@implementation MYViewControllerAnimatedTransitioningV2

#pragma mark UIViewControllerAnimatedTransitioning

- (NSTimeInterval)transitionDuration:(id<UIViewControllerContextTransitioning>)transitionContext{
    return kTransitionDuration;
}

- (void)animateTransition:(id<UIViewControllerContextTransitioning>)transitionContext{
    switch (_type) {
        case UIViewControllerPresentTransitionTypePresent:
            [self animatePresentTransition:transitionContext];
            break;
        case UIViewControllerPresentTransitionTypeDismiss:
            [self animateDissmissTransition:transitionContext];
            break;
        case UIViewControllerPresentTransitionTypePush:
            [self animatePushTransition:transitionContext];
            break;
        case UIViewControllerPresentTransitionTypePop:
            [self animatePopTransition:transitionContext];
        default:
            break;
    }
}


- (void)animatePushTransition:(id<UIViewControllerContextTransitioning>)transitionContext{
    UIViewController *fromViewController= [transitionContext viewControllerForKey:UITransitionContextFromViewControllerKey];
    UIViewController *toViewController = [transitionContext viewControllerForKey:UITransitionContextToViewControllerKey];

    UIView *containView = [transitionContext containerView];

    UIView *tabbar = fromViewController.tabBarController.tabBar;
    if(tabbar != nil && ![containView.subviews containsObject:tabbar]){
        self.tabBarSuperview = tabbar.superview;
        tabbar.frame = CGRectMake(0, [UIScreen mainScreen].bounds.size.height - 49, [UIScreen mainScreen].bounds.size.width, 49);
        [containView addSubview:tabbar];
    }

    [containView addSubview:toViewController.view];

    CGRect rect = [transitionContext finalFrameForViewController:toViewController];
    rect.origin.x = [[UIScreen mainScreen] bounds].size.width;

    CGFloat offset = 0;
    rect.origin.y = offset;
    toViewController.view.frame = rect;

    toViewController.view.layer.shadowColor = [UIColor blackColor].CGColor;//添加阴影
    toViewController.view.layer.shadowOffset = CGSizeMake(0, 0);
    toViewController.view.layer.shadowRadius = 3;
    toViewController.view.layer.shadowOpacity = 0.8;
    toViewController.view.layer.shadowPath = [UIBezierPath bezierPathWithRect:CGRectMake(0, 0, toViewController.view.frame.size.width, toViewController.view.frame.size.height)].CGPath;

     [UIView animateWithDuration:[self transitionDuration:transitionContext] delay:0 options:UIViewAnimationOptionCurveEaseInOut animations:^{
        CGRect frame = rect;
        frame.origin.x = 0;
        frame.origin.y = offset;
        toViewController.view.frame = frame;

        frame.origin.x = - [[UIScreen mainScreen] bounds].size.width/3;
        frame.origin.y = offset;
        fromViewController.view.frame = frame;

         if(tabbar != nil && [containView.subviews containsObject:tabbar]){
            tabbar.frame_x =  [[UIScreen mainScreen] bounds].size.width * 2 / 3;
         }
    } completion:^(BOOL finished) {
        BOOL isCanceled = [transitionContext transitionWasCancelled];
        [transitionContext completeTransition:!isCanceled];
        if(!isCanceled){
        }
    }];
}
```

```
- (void)animatePopTransition:(id<UIViewControllerContextTransitioning>)transitionContext{
  UIViewController *toViewController = [transitionContext viewControllerForKey:UITransitionContextToViewControllerKey];
  UIView *fromViewControllerView = [transitionContext viewForKey:UITransitionContextFromViewKey];//No:1 -> No:3

  UIView *containView = [transitionContext containerView];

  [containView addSubview:toViewController.view];//No:2

  UIView *tabbarview = toViewController.tabBarController.tabBar;
  if(tabbarview != nil && [containView.subviews containsObject:tabbarview]){
    [containView bringSubviewToFront:tabbarview];
  }else{
    self.tabBarSuperview = tabbarview.superview;
    tabbarview.frame = CGRectMake([[UIScreen mainScreen] bounds].size.width * 2 / 3, [UIScreen mainScreen].bounds.size.height - 49, [UIScreen mainScreen].bounds.size.width, 49);
    [containView addSubview:tabbarview];
  }

  [containView bringSubviewToFront:fromViewControllerView];

  CGRect rect = [transitionContext finalFrameForViewController:toViewController];
  rect.origin.x = -[[UIScreen mainScreen] bounds].size.width/3;
  toViewController.view.frame = rect;

  if(tabbarview != nil && [containView.subviews containsObject:tabbarview]){
    tabbarview.frame_x = - [[UIScreen mainScreen] bounds].size.width/3;
  }

  CGFloat offset = 0;

  fromViewControllerView.layer.shadowColor = [UIColor blackColor].CGColor;//(2)pop前把(1)操作取消的阴影加上



  __weak typeof(self) weakSelf = self;
  [UIView animateWithDuration:[self transitionDuration:transitionContext] delay:0 options:UIViewAnimationOptionCurveEaseInOut animations:^{
    CGRect frame = rect;
    frame.origin.x = 0;
    frame.origin.y = offset;
    toViewController.view.frame = frame;

    if(tabbarview != nil && [containView.subviews containsObject:tabbarview]){
      tabbarview.frame_x = 0;
    }

    frame.origin.x = [[UIScreen mainScreen] bounds].size.width;
    frame.origin.y = offset;
    fromViewControllerView.frame = frame;
  } completion:^(BOOL finished) {
    BOOL isCanceled = [transitionContext transitionWasCancelled];
    [transitionContext completeTransition:!isCanceled];
    if(!isCanceled){
      toViewController.view.layer.shadowColor = [UIColor clearColor].CGColor;//(1)解决pop结束阴影还在
      if(tabbarview != nil && [containView.subviews containsObject:tabbarview]){
        tabbarview.frame = CGRectMake(0, [UIScreen mainScreen].bounds.size.height - 49, [UIScreen mainScreen].bounds.size.width, 49);
        [weakSelf.tabBarSuperview addSubview:tabbarview];
      }
    }else{//手势取消，containtView结构复位
      if(tabbarview != nil && [containView.subviews containsObject:tabbarview]){
        [containView bringSubviewToFront:tabbarview];
      }
      [containView bringSubviewToFront:fromViewControllerView];
    }
  }];
}
```

备注：封装基本结束，具体封装的使用方法运行下演示项目就可以看到了。我们的目的是一行代码能解决的事，多敲一行都不行！后续，会抽时间来对此封装做详细说明，暂时就先这样啦~