**四川师范大学本科毕业设计**

|  |
| --- |
| 基于iOS的美妆电商类APP实现 |

|  |  |
| --- | --- |
| **学生姓名** | **＊＊＊** |
| **院系名称** | **计算机科学学院** |
| **专业名称** | **软件工程** |
| **班 级** | **2＊＊＊级＊班** |
| **学 号** | **＊＊＊** |
| **指导教师** | **李贵洋** |
| **完成时间** | **2016年5月6日** |

**基于iOS的美妆电商类APP实现**

学生：＊＊＊ 指导教师：李贵洋

内容摘要：此为内容摘要，大概300字

关键词：iOS开发 移动智能客户端 电子商务

**THE IMPLEMENTATION OF ELECTRICITY BUSINESS APPLICATION FOR IOS**

**Abstract:** say something lalalalala

**Key Words: iOS development Mobile intelligent terminal Electricity business**

# 目录

1. 概述 ………………………………………………………………………………
   1. 研究的目的和意义 ………………………………………………………
   2. 研究的背景 ………………………………………………………………
      1. 国内现状 …………………………………………………………
      2. 国外现状 …………………………………………………………
   3. 主要贡献 …………………………………………………………………
   4. 文章的结构 ………………………………………………………………

2. 预备知识及原理说明 ……………………………………………………………

2.1 Cocoa Touch …………………………………………………………………

2.2开发语言及开发工具 ………………………………………………

2.3设计开发模式MVC …………………………………………

2.4 依赖管理工具CocoaPods

2.5 代码管理工具 Git

2.4 iOS 第三方库 ………………………………………………………………

2.5 版本控制软件git …………………………

3. 软件系统分析 ………………………………………….

4．软件设计和实现 ………………………………………

5. 基于iOS的电商美妆app系统测试和运行 ……………………………………

6. 成果与展望 ………………………………………………………………………

7. 致谢 ………………………………………………………………………………

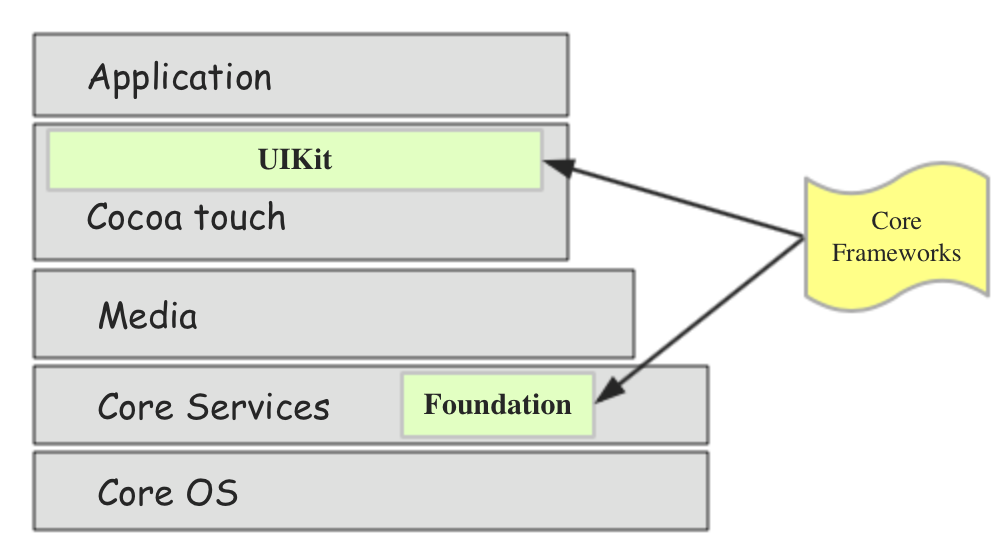
参考文献 ………………………………………………………………………………

**基于iOS的美妆电商类APP实现**

# 1 概述

# 2预备知识及原理说明

# 2.1 Cocoa Touch

****

**iOS 系统架构**

Cocoa不是一种编程语言（它可以运行多种编程语言），它也不是一个开发工具（通过命令行我们仍然可以开发Cocoa程序），它是创建Mac OS X和IOS程序的原生面向对象API，为这两者应用提供了编程环境。

Cocoa Touch 是针对于iOS的应用开发环境，它包括Objective-C 运行环境和两个核心框架：foundation框架和ApplicationKit（UIKit）框架。框架的功能类似于动态库，既可以在运行时动态的载入应用程序的地址空间，但框架作为一个对计算机的捆绑，而非独立文件，除了可执行代码外，也包含了资源，头文件和文档。

foundation框架作为通用的面向对象函数库，主要定义了一些基础类，提供了字符串，数值的管理，容器及其枚举，以及一些其他的与图形用户界面 没有直接关系的功能；UIKit框架提供了在屏幕上绘制的机制，捕获事件，和创建通用用户界面元素。UIKit也通过管理显示在屏幕上的组件来组织复杂的项目。使用UIKit可以：构建和管理你的用户界面，捕获触摸和基于移动的事件，呈现文字和web内容，优化你的多任务程序，创建定制的用户界面元素。

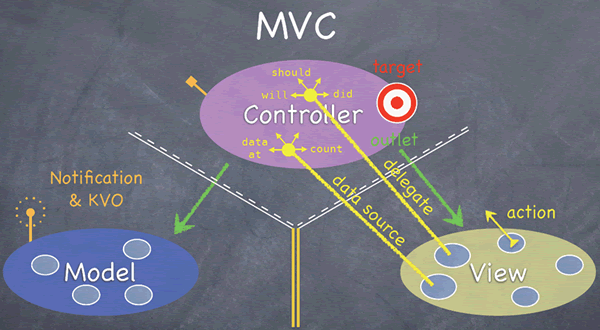
**2.2 开发语言及开发工具**

20世纪八十年代初，Brad Cox在其公司Stepstone发明Objective-C，它以一种叫做SmallTalk-80的语言为基础。Objective-C建立在C语言之上，意味着它是在C语言基础上添加了扩展而创造出来的能够创建和操作对象的一门新的程序设计语言。对Objective-C最主要的描述是他1986年出版的《 Object-oriented Programming, An Evolutionary Approach》。1988年，NeXT Software公司获得了Objective-C语言的授权，并开发出了Objective-C的语言库和一个名为NEXTSTEP的开发环境。1992年，自由软件基金会的GNU开发环境增加了对Objective-C的支持。1994年，NeXT Computer公司和升阳微系统（Sun Microsystem）联合发布了一个针对 NEXTSTEP系统的标准典范，名为OPENSTEP。OPENSTEP在自由软件基金会的实现名称为GNUstep。1996年12月20日，苹果公司宣布收购NeXT Software公司，NEXTSTEP/OPENSTEP环境成为苹果操作系统下一个主要发行版本OS X的基础。这个开发环境的该版本被苹果公司称为Cocoa。（互动百科）

苹果公司于2008年3月6日发布了iPhone和iPod touch的应用程序开发包,其中包括Xcode开发工具、iPhone SDK 和iPhone手机模拟器。第一个Beta版本是iPhone SDK 1.2b1(build 5A147p),发布后立 能使用, 是同时 出 的App Store所需要的 件更新 到2008年7月11日 发布。编写本书时,iOS SDK 7.0.4版本 经发布。  iOS开发工具主要是Xcode。自从Xcode 3.1发布以后,Xcode 成为iPhone 件开发工具包的开发 。Xcode 可以开发Mac OS X和iOS应用程序,其版本是与SDK相 对应的。例如,Xcode 3.2.5与iOS SDK 4.2对应,Xcode 4.1 与iOS SDK 4.3对应,Xcode 4.2与iOS SDK 5对应,Xcode 4.5和Xcode 4.6与iOS SDK 6对应,Xcode 5与iOS SDK 7 对应。  在Xcode 4.1之前,还有一个配 使用的工具Interface Builder, 是Xcode 件的一部分,用来设计 体和视 图,通过 可以 所见 所得 地 控件并定 事件等,其数据以XML的形式 存储在xib文件中。在Xcode 4.1 之后, Interface Builder成为了Xcode的一部分,与Xcode集成在一 。 （iOS开发指南：从零基础到App Store上架（第2版 ）

**2.3设计开发模式MVC**

MVC，全称是 Model View Controller，是模型 (model)－视图 (view)－控制器 (controller) 的缩写。它表示的是一种常见的客户端软件开发框架。MVC 的概念最早出现在二十世纪八十年代的施乐帕克实验室中，当时施乐帕克为 Smalltalk 发明了这种软件设计模式。现在，MVC 已经成为主流的客户端编程框架，在 iOS 开发中，系统为我们实现好了公共的视图类：UIView，和控制器类：UIViewController。大多数时候，我们都需要继承这些类来实现我们的程序逻辑，因此，我们几乎逃避不开 MVC 这种设计模式。



iOS系统下的MVC关系图

Paul Hegarty，斯坦福大学公开课：iOS 7应用开发[EB/OL]. 网易公开课.2014.

Controller能够访问Model和View，Model和View不能互相访问；当View与用户交互产生事件时，使用target-action方式来处理；当View需要处理一些特殊UI逻辑或获取数据源时，通过delegate或data source方式交给Controller来处理；Model不能直接与Controller通信，当Model有数据更新时，可以通过Notification或KVO (Key Value Observing)来通知Controller更新View。

另外由MVC还衍生出了一些其他的架构：MVVM，MVCS，VIPER等。

MVVM是基于胖model的架构思路建立，在胖model中拆出两部分：model和viewModel，viewModel让Controller只需关注于数据调配，它去负责数据加工并通过通知机制让view响应viewModel的改变，总之不管是胖model还是viewModel都是给controller减负。

（PS：胖model：即包含了部分业务逻辑，controller从其拿到数据之后不用做额外操作或很少操作就能使用，瘦model：只负责数据的表达，然后配套helper类或方法来对弱业务做抽象，其他一律扔给controller。独立性强。但一定程度上违背了DRY，即Don’t repeat yourself）

MVCS是从controller中把负责数据存储部分提取出来，交给另外一个对象去做，这个对象就是store。

VIPER，即视图 (View)，交互器 (Interactor)，展示器 (Presenter)，实体 (Entity) 以及路由 (Routing)。将一个应用程序的逻辑结构划分为不同的责任层。这使得它更容易隔离依赖项 (如数据库)，也更容易测试各层间的边界处的交互。

**2.4 依赖管理工具CocoaPods**

随着iOS开发者的增多，业界也出现了为iOS程序提供依赖管理的工具，经过多年发展，现在已经成为iOS开发事实上的依赖管理标准工具。开发iOS项目不可避免地要使用第三方开源库，CocoaPods使得我们可以节省设置和更新第三方开源库的时间。

引用第三方库，一般需要：把这些第三方开源库的源代码复制到项目中，或设置成git的submodule；需要手工地将这些开源库所依赖的系统框架一一添加到项目依赖中，比如一个网络库需要添加的框架有CFNetwork, SystemConfiguration, MobileCoreServices, CoreGraphic, zlib; 对于某些开源库需要设置-licucore或者-fno-objc-arc等编译参数；最后还得管理这些依赖包的更新。

使用CocoaPods之后，我们只需要将用到的第三方开源库的源码放到一个名为Podfile的文件中，然后执行Pod install。CocoaPods就会自动将这些第三方开源库下载下来，并且为我们的工程设置好相应的系统依赖和编译参数，执行pod update便可更新指定的第三方库。《iOS开发进阶》唐巧著. 北京：电子工业出版社，2015.1 ISBN 978-7-121-24745-3

**2.5版本控制软件git**

git是一个免费的、分布式的版本控制工具，或是一个强调了速度快的源代码管理工具。Git最初被Linus Torvalds开发出来用于管理Linux内核的开发。每一个Git的工作目录都是一个完全独立的代码库，并拥有完整的历史记录和版本追踪能力，不依赖于网络和中心服务器。git和其他版本控制系统（如CVS）有不少的差别，git本身关心文件的整体性是否有改变，但多数的CVS或Subversion系统则在乎文件内容的差异。因此git更像一个文件系统，直接在本机上获取数据，不必连接到主机端获取数据。

2.6 关键三方库

2.6.1自动布局三方库：PureLayout.h

PureLayout是 iOS & OS X Auto Layout的终极 API——非常简单，又非常强大。PureLayout通过一个全面的Auto Layout API扩展了UIView/NSView, NSArray 和 NSLayoutConstraint，仿照苹果自身的框架。

基础用法：

|  |
| --- |
| 1.view相对于父容器间距的位置 |
| [view autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:10];//相对于父容器顶部距离10 [view autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:10];//相对于父容器左部距离10 [view autoPinEdgeToSuperviewEdge:ALEdgeRight withInset:10];//相对于父容器右部距离10 [view autoPinEdgeToSuperviewEdge:ALEdgeBottom withInset:10];//相对于父容器底部距离10  值得注意的是Purelayout对UILabel做了一些人性化的处理： 在有的国家地区文字是从右至左的，以下代码就是将label的起始位置距离父容器10  [label autoPinEdgeToSuperviewEdge:ALEdgeLeading withInset:10]; |

|  |
| --- |
| 2.相对于父容器的中心位置： |
| [view autoCenterInSuperview];//view在父容器中心位置 [view autoAlignAxisToSuperviewAxis:ALAxisHorizontal];//view在父容器水平中心位置 [view autoAlignAxisToSuperviewAxis:ALAxisVertical];//view在父容器垂直中心位置 |

|  |
| --- |
| 3.设置大小 |
| [view autoSetDimensionsToSize:CGSizeMake(300, 300)];//设置view的大小为300\*300 [view autoSetDimension:ALDimensionHeight toSize:300];//设置view的高度为300 [view autoSetDimension:ALDimensionWidth toSize:300];//设置view的宽度为300 |

|  |
| --- |
| 4.相对位置 |
| NSLayoutRelation是一个枚举类型：  typedef NS\_ENUM(NSInteger, NSLayoutRelation) { NSLayoutRelationLessThanOrEqual = -1, NSLayoutRelationEqual = 0, NSLayoutRelationGreaterThanOrEqual = 1, };  见名知意，你懂的。  [view1 autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:view2 withOffset:20 relation:NSLayoutRelationEqual];//view1的顶部在view2的底部的20像素的位置 |

|  |
| --- |
| 5.中心对齐 |
| [view1 autoAlignAxis:ALAxisVertical toSameAxisOfView:view2];//view1相对于view2保持在同一个垂直中心上  view1相对于view2保持在同一个垂直中心上 |

|  |
| --- |
| 6.相对大小 |
| [view1 autoMatchDimension:ALDimensionWidth toDimension:ALDimensionWidth ofView:view2];  view1的宽度和view2的宽度相等 |

2.6.2 刷新组件：SVPullToRefresh

SVPullToRefresh是iOS上实现下拉刷新和上提加载的一个组件。使用场景仅限UIScrollView，只需要一行代码，就能添加下拉刷新的功能。

[tableView addPullToRefreshWithActionHandler:^{ // 预加载数据到dataSource, 向tableview插入cell // 完成时调用[tableView.pullToRefreshView stopAnimating] }];

### PullToRefresh实现机制

PullToRefresh就是在UIScrollView上面add了一个subview：PullToRefreshView。这个PullToRefreshView就是一个普通的View，宽度和UIScrollView相同，高度可以自定义，默认是60。

PullToRefreshView有四种状态：

 SVPullToRefreshStateStopped：刷新停止状态，还没有触发刷新操作。

 SVPullToRefreshStateTriggered：刷新操作被触发，但还没有开始发送请求（只要用户手还没松开，不会发送请求）

 SVPullToRefreshStateLoading：已经发送请求，但数据还没回来，这时菊花一直在转。这时候UITableView的contentInset已经修改，使PullToRefreshView可见。

 SVPullToRefreshStateAll：初始化的时候用的。

如果不需要自定义，PullToRefreshView内置了一个默认版本，包含箭头、title、subtitle、time。

自定义也可以，在PullToRefreshView的viewForState数组中加入三中状态的视图，然后就可以显示自定义的UI了。

PullToRefreshView是UIScrollView的一个属性，这个属性是通过关联的方式绑定的。当用户滚动UIScrollView的时候，ContentOffset会发生变化，这时候UIScrollView的观察者PullToRefreshView就会收到通知，然后根据PullToRefreshView自身的状态和ContentOffset决定是否切换状态，并执行相应动画。

下拉刷新动画分成两个阶段：

* 第一个阶段是还没进入加载阶段，这时候随着下拉的高度，图片周围一圈扇形区域逐渐变大。
* 第二个阶段是加载过程中，中间的图片开始旋转，同时下面伸出一根点段不断伸缩，直到加载结束。

上滑动画是下拉动画的简化版，就是下拉动画的第二个阶段；  
另外可以用一个属性来控制当没有更多数据的时候，显示没有更多数据。

2.6.3正在加载三方库：MBProgressHUD

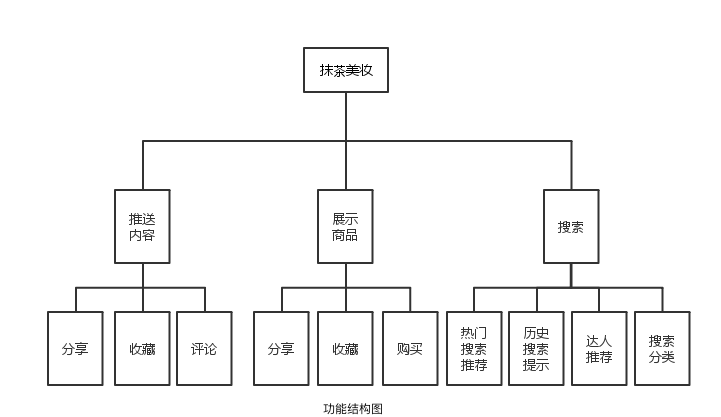
很多时候，我们进入页面的时候，因为程序正在读取数据，页面呈现出一片空白。这会引起用户的焦虑，造成不好的用户体验。这个时候，如果能用一个过渡，便会缓解这种状况。**MBProgressHUD**就是为了满足这种需求开发出来的第三方工具，使用 MBProgressHUD 在空白期出现一个转动的小菊花，用来过滤。

|  |
| --- |
| 基础使用方法： |
| @property (nonatomic,strong) MBProgressHUD \*HUD;  - (MBProgressHUD \*)HUD{  if (!\_HUD) {  \_HUD = [[MBProgressHUD alloc]initWithView:self.view];  \_HUD.labelText = @”加载中…”;  }  return \_HUD;  }  [\_HUD show:YES];//显示  [\_HUD hide:YES];//隐藏 |

# 3 软件分析

**3.1 需求分析**

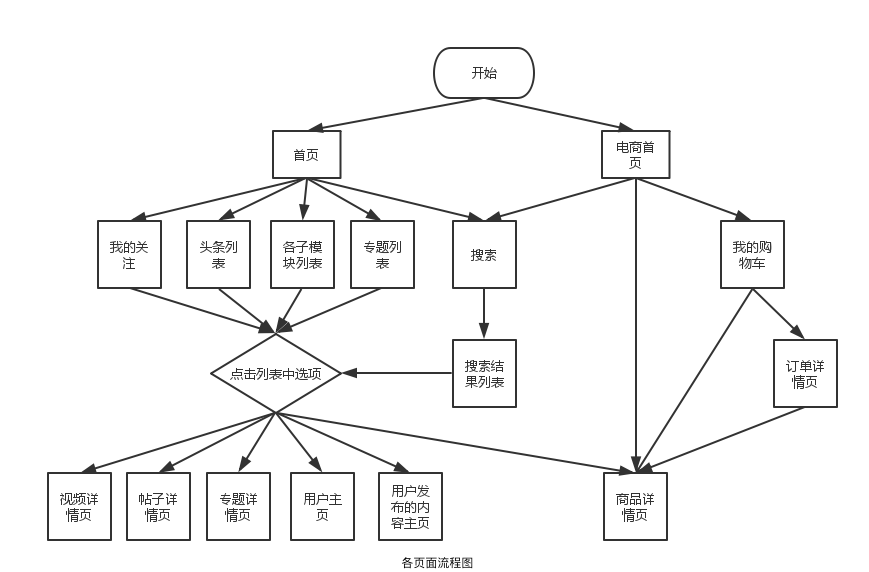
为了实现两大需求，第一是实现动态显示从服务端获取的指定信息，如视频，帖子，专题等，并支持和用户间的互动，比如收藏内容，关注发布者，评论信息等；第二是线上商品销售，主要针对美妆类商品，支持动态展示商品，加入购物车，下单结算等电商模式；编写一个基于iOS的客户端。



具体产品功能需求如下：

1. 对于新用户，有个用法引导，对于老用户升级到新版本后，针对新功能也有个引导提示，所有引导只显示一次，除非卸载后重新安装。
2. 首页模块中，顶部导航栏有支持横滑的选项卡，选项卡元素包括关注、头条、护肤、彩妆、时尚、专题，除了关注，头条，专题这三块是固定显示之外，其他选项内容是动态从服务端获取。每个选项卡对应不同页面，每个页面显示样式不同。打开软件进入首页后默认主页面，即跳转到哪个选项由服务端确定。
3. 首页所有页面支持下拉刷新以及上拉加载更多，进入主页面后，第一次以及超出指定时间间隔后滑动至或点击选项卡跳转至其它页面后支持自动刷新。
4. 首页中的关注页面，主要显示用户所关注的用户发布的内容，以及显示推荐用户，便于用户关注。如果进入该页面后，用户没登陆，得提醒用户登陆，并提供入口。
5. 首页中的头条页面，也是核心页面，它是具体所发布的信息的入口，它主要由两大显示块组成，一是顶部的banner,即支持横滑，也支持自动轮播的推荐内容入口，点击不同内容跳到不同的详情页；其他的就是以表视图形式展示信息，并支持不同类型的cell显示，其中视频类的cell点击播放按钮要支持在当前界面播放，当视频滑出屏幕后要关闭视频，针对视频和帖子类型的cell会有发布者的信息，如果用户没关注该发布者，则提供关注按钮，用户点击“关注”即可关注该用户，点击发布者头像会跳到发布者主页。
6. 首页其他各选项即是所有信息的分类显示，其中专题分类以feed流表视图的形式展示，其他的分类以瀑布流的形式展示。点击后都会跳转至对应详情页。
7. 首页每个cell里可能会包含推荐的商品，如果有则有相应按钮，点击弹出弹层，展示商品列表，如果该商品有库存则点击商品跳转至商品详情页，如果无货，则提供“种草”，即可加入心愿单。
8. 专题详情页所包含的内容有封面图，标题，分享入口，发布者信息，具体内容，各评论信息。其中服务端返回的内容部分包括：图片文字，推荐的商品信息（可加入心愿单或加入购物车或跳至详情页购买），视频（支持当前播放），其他相关专题入口。
9. 电商模块中主页模式与首页有点相似，只是是各商品的展现，点击具体商品跳至相应商品详情页。
10. 商品详情页从上到下包括的内容为，支持横滑的商品图片及视频，商品名，商品售价，简介，部分评论，详细信息，其他专题及商品推荐。以及各入口，包括分享，客服入口，查看购物车，加入购物车。点击加入购物车要判断该商品是否有多种规格，若无则直接加入购物车，若有，则弹出弹层，供用户选择，其中不能点击没库存的规格，选择完后再加入购物车。
11. 搜索模块，搜索的入口在首页导航栏上，即选项卡的右边。搜索页面中：导航栏中有搜索输入框，可搜索出发布的各种信息，下边界面上则有四个版块，热门搜索：提供热门的搜索标签，可直接点击进行搜索该标签内容，历史搜索：所有的历史搜索词都以标签形式展示，用户可直接点击搜索，另外有删除历史搜索纪录按钮，若没有历史搜索则无该版块，推荐用户：点击可进入相应用户所有发布内容的界面，其他推荐：该版块下把所有搜索内容分成有五类：妆品分类，妆品功效，基础彩妆，基础护肤，化妆风格，点击跳转的页面把这五类又细分了很多块，并且点击不同类，跳转的页面对应相应的分类。

**3.2 软件流程**



**软件各界面关系，以及跳转关系图（部分）**

**4 软件设计**

**4.1 基于MVC的架构设计**

**app设计中，除了第三库，工具类的引用，主要把程序代码分为了三个部分，模型（model），视图控制器（controller），视图（view），部分把controller里的内容分了个VM出来，以减轻视图控制器的负担。**

****

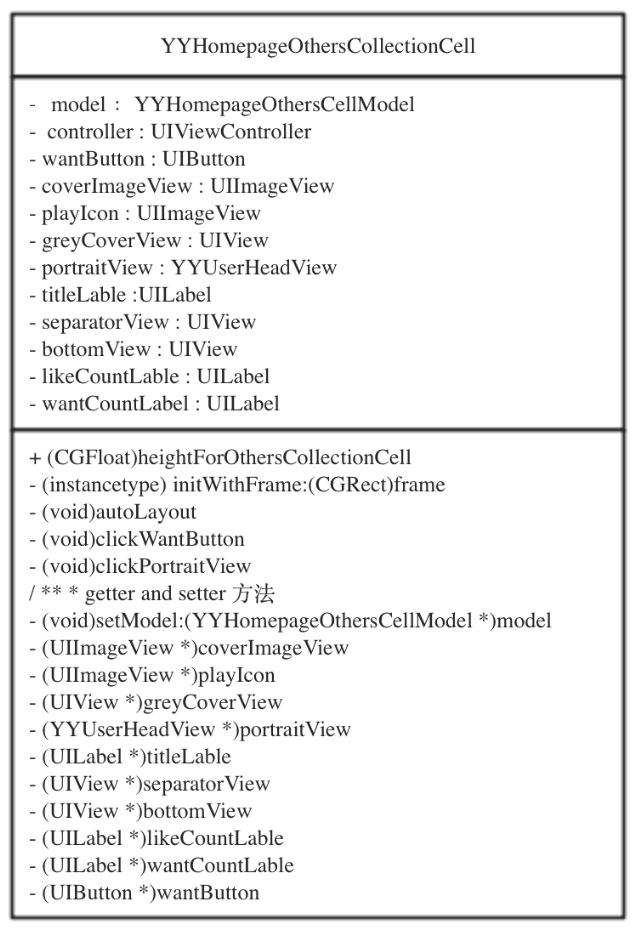
**4.2 关键类**

模型代表类



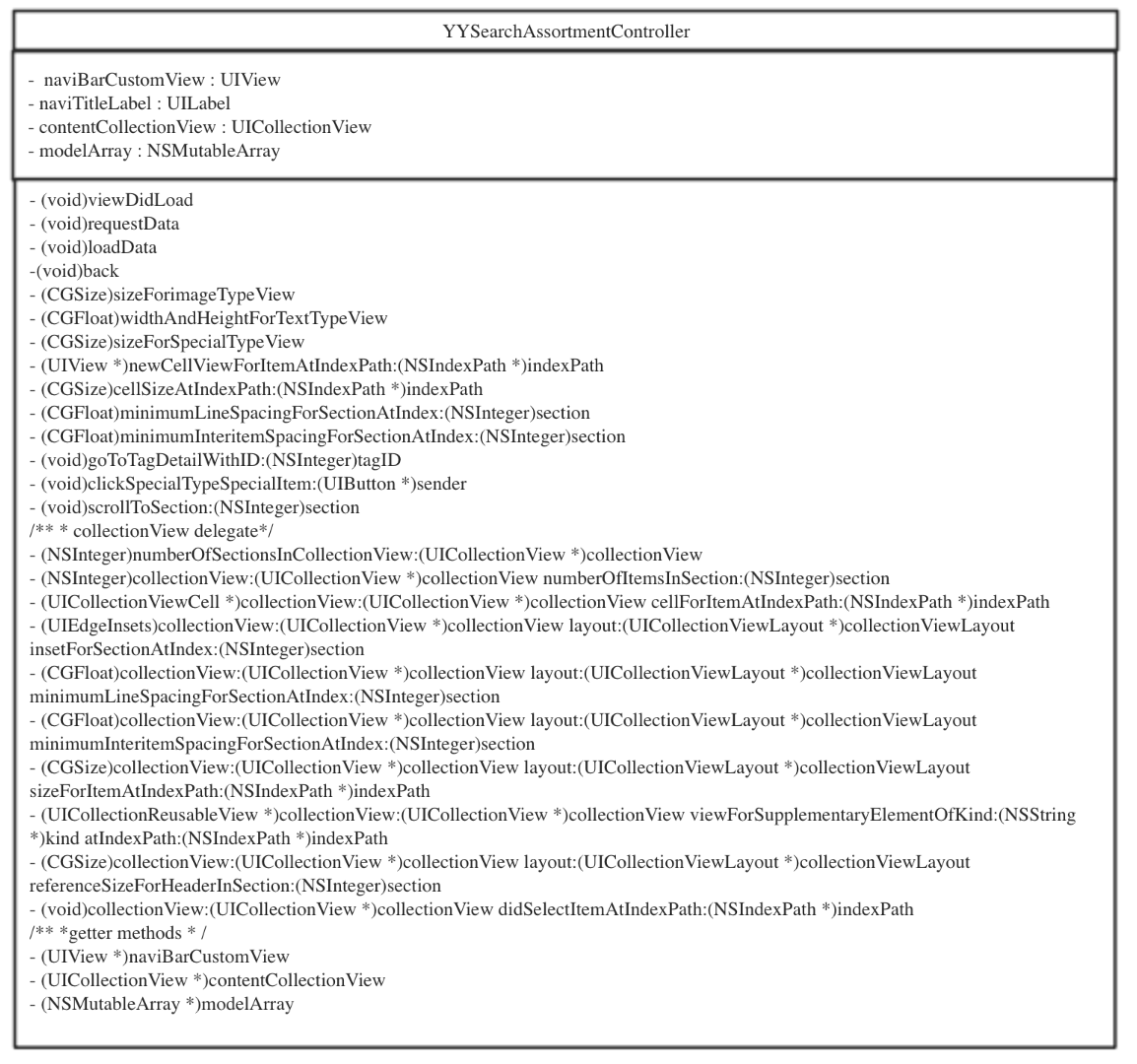
图x-x 为首页头条中视频帖子类cell的模型类图，类方法作用是转化字典数据，因为从服务器中返回的数据默认为字典类型。

view代表类图



图x-x 为首页子分类模块中的cell视图类图

视图控制器 代表类图：



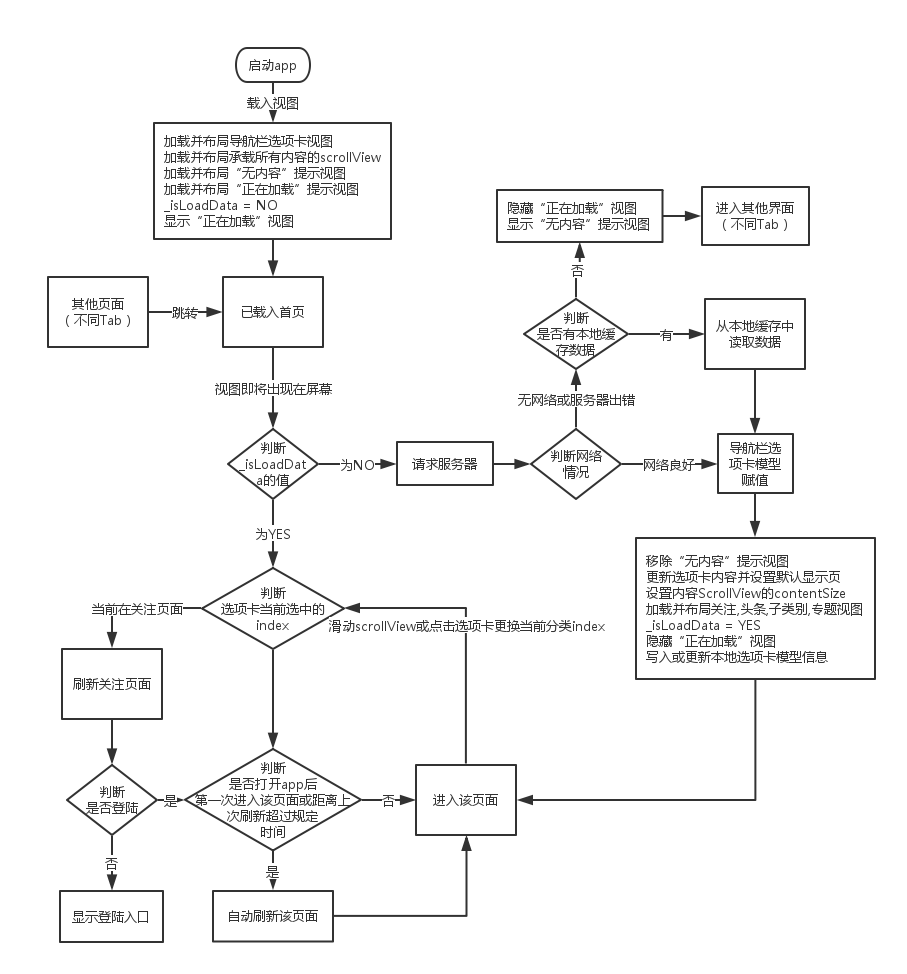
**4.3 重要模块详细设计**

4.3.1 首页导航栏及各页面切换

打开app进入首页或者是从底部tab栏选择到首页，会处理几个逻辑：网络打开关闭状态和登陆状态的组合以及其切换时间组合情况下的逻辑处理，各分类页面视图加载数据或更新数据逻辑处理。



效果图



流程图

|  |
| --- |
| 导航栏逻辑 |
| 注：该逻辑包含以下四个函数   * (void)viewDidLoad ； * (void)viewWillAppear:(BOOL)animated； * (void)requestData；   - (void)loadMainView； |
| #pragma mark - life cycle  - (void)viewDidLoad {  [super viewDidLoad];  [self.view addSubview:self.naviBarCustomView];  [self.view addSubview:self.customContentView];  [self.view addSubview:self.defaultView];  [self.view addSubview:self.HUD];  [\_HUD show:YES];  [\_defaultView autoAlignAxisToSuperviewAxis:ALAxisVertical];  [\_defaultView autoAlignAxisToSuperviewAxis:ALAxisHorizontal];  \_isLoadData = NO;  \_tempPlayerView = [YYPlayerView sharedInstance];  }  - (void)viewWillAppear:(BOOL)animated{  [super viewWillAppear:animated];  //在没网状态下加载应用：每次进入首页重新加载数据  if (!\_isLoadData) {  [self requestData];  }  //处在关注界面下切换登录状态后的处理  if (self.topTabView.currentItemIndex == tabType\_subscription) {  [\_subscriptionView refreshView];  }  }  #pragma mark - private methods  - (void)requestData{  \_\_weak typeof(self) weakSelf = self;  [YYHttpUtil httpPostWithAuthInfo:nil withUrl:@"/front/hometab/list" withHttpSuccessBlock:^(id data) {  [weakSelf.topTabList removeAllObjects];  for (NSDictionary \*dic in data) {  [weakSelf.topTabList addObject:[YYHomepageTabList parseData:dic]];  }  [weakSelf loadMainView];    NSDictionary \*tempDic=[NSDictionary dictionaryWithObject:data forKey:@"dataKey"];  [YYUtil saveDataDict:(NSDictionary \*)tempDic withLocalFileName:dataFileName];  [\_HUD hide:YES];  } withHttpExceptionBlock:^(NSUInteger code, NSString \*msg) {  NSDictionary \*localData = [YYUtil fetchDataDictWithLocalFileName:dataFileName];  if (localData) {  [weakSelf.topTabList removeAllObjects];  for (NSDictionary \*dic in [localData valueForKey:@"dataKey"]) {  [weakSelf.topTabList addObject:[YYHomepageTabList parseData:dic]];  }  [self loadMainView];  }  if (!\_isLoadData) {  \_defaultView.hidden = NO;  }  [\_HUD hide:YES];  } withFailedBlock:^(AFHTTPRequestOperation \*operation, NSError \*error) {  NSDictionary \*localData = [YYUtil fetchDataDictWithLocalFileName:dataFileName];  if (localData) {  [weakSelf.topTabList removeAllObjects];  for (NSDictionary \*dic in [localData valueForKey:@"dataKey"]) {  [weakSelf.topTabList addObject:[YYHomepageTabList parseData:dic]];  }  [self loadMainView];  }  if (!\_isLoadData) {  \_defaultView.hidden = NO;  }  [\_HUD hide:YES];  }];  }  - (void)loadMainView{  [\_defaultView removeFromSuperview];  [self.topTabView updateWithTabList:\_topTabList];  int index = 0;  for (YYHomepageTabList \*tempTab in \_topTabList){  if([tempTab.isDefaultPage boolValue])  {  self.topTabView.currentItemIndex = index;  break;  }  index++;  }    [\_customContentView setContentSize:CGSizeMake(kDeviceWidth\*[self.topTabList count], CONTENT\_HEIGHT\_VISIBLE)];  [self.customContentView addSubview:self.subscriptionView];  [self.customContentView addSubview:self.hotView];  [self.customContentView addSubview:self.otherListsView];  [self.customContentView addSubview:self.subjectsListView];  [\_otherListsView setUpWithTabArray:\_topTabList];  [\_customContentView setContentOffset:CGPointMake(self.topTabView.currentItemIndex\*kDeviceWidth, 0)];  \_isLoadData = YES;  } |

**4.3.2 首页中头条分类下逻辑**

****

****

效果图

|  |
| --- |
| 头条页各种类型cell处理 |
| 注：   1. YYTableView 只是在系统的UITableView上封装了一个返回顶部的按钮，其他代理方法等与UITableView相同 2. 以下为两个主要的方法：一是表视图的创建，即getter方法；二是表视图代理方法中返回cell的方法   - (YYTableView \*)tableView;  - (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath |
| - (YYTableView \*)tableView{  if (!\_tableView) {  \_tableView = [[YYTableView alloc]init];  \_tableView.delegate = self;  \_tableView.dataSource = self;  \_tableView.backgroundColor = mochaViewBackgroundColor;  \_tableView.separatorStyle = UITableViewCellSelectionStyleNone;  [\_tableView registerClass:[YYHomepageHotCell class] forCellReuseIdentifier:cellIdentifier];    \_\_weak typeof(self) wSelf = self;  [\_tableView addPullToRefreshWithActionHandler:^{  [wSelf loadData];  }];  [\_tableView addInfiniteScrollingWithActionHandler:^{  [wSelf loadMoreData];  }];  [\_tableView.pullToRefreshView setTitle:refreshStateStoppedString forState:SVPullToRefreshStateStopped];  [\_tableView.pullToRefreshView setTitle:refreshStateTriggeredString forState:SVPullToRefreshStateTriggered];  [\_tableView.pullToRefreshView setTitle:refreshStateLoadingString forState:SVPullToRefreshStateLoading];  }  return \_tableView;  }  - (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath {  if (indexPath.section == 0) {  UITableViewCell \*cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:@"titleCell"];  [cell.contentView addSubview:self.bannerView];  cell.contentView.backgroundColor = mochaViewBackgroundColor;  //让bannerView超出父视图一点，盖住下面cell的分割线  [\_bannerView autoPinEdgeToSuperviewEdge:ALEdgeTop];  [\_bannerView autoPinEdgeToSuperviewEdge:ALEdgeLeft];  [\_bannerView autoSetDimension:ALDimensionWidth toSize:kDeviceWidth];  bannerHeightConstr = [\_bannerView autoSetDimension:ALDimensionHeight toSize:bannerHeight + separatorHeight];  cell.backgroundColor = [UIColor clearColor];  cell.contentView.backgroundColor = [UIColor clearColor];  cell.selectionStyle = UITableViewCellSelectionStyleNone;  return cell;  }else{  YYHomepageHotCellModel \*model = [self.dataSource objectAtIndex:indexPath.row];  if (model.sourceType == ShopVideoListDataType\_Subscription) {//关注、订阅  YYShopSubscriptionCell \*cell = [tableView dequeueReusableCellWithIdentifier:@"subscriptionCell"];  if (!cell) {  cell = [[YYShopSubscriptionCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:@"subscriptionCell"];  }  cell.controller = self.controller;  [cell updateCellwithData:model];  return cell;  }else if (model.sourceType == ShopVideoListDataType\_Ad){  YYShopTouTiaoAdCell \*cell = [tableView dequeueReusableCellWithIdentifier:@"adCell"];  if (!cell) {  cell = [[YYShopTouTiaoAdCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:@"adCell"];  }  [cell updateCellWithData:model];  return cell;  }else if(model.sourceType == ShopVideoListDataType\_VideoSubject || model.sourceType == ShopVideoListDataType\_Album){//专题  YYSubjectCell \*cell = [tableView dequeueReusableCellWithIdentifier:@"videoSubjectCell"];  if (!cell) {  cell = [[YYSubjectCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:@"videoSubjectCell"];  }  cell.homepageModel = model;  cell.controller = self.controller;  return cell;  }else if(model.sourceType == ShopVideoListDataType\_Video ||model.sourceType == ShopVideoListDataType\_Post || model.sourceType == ShopVideoListDataType\_Teach){  YYHomepageHotCell \*cell = [tableView dequeueReusableCellWithIdentifier:cellIdentifier forIndexPath:indexPath];  if (!cell) {  cell = [[YYHomepageHotCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:cellIdentifier];  }  cell.model = model;  cell.controller = self.controller;  cell.playerView = self.playerView;    \_\_weak typeof(cell) weakCell = cell;  \_\_weak typeof(model) weakModel = model;  [weakCell setClickPlayButtonBlock:^{  [YYHttpUtil httpPostWithAuthInfo:[NSDictionary dictionaryWithObjectsAndKeys:weakModel.cusID,@"videoID", nil] withUrl:@"/youpin/video/viewcount/add" withHttpSuccessBlock:^(id data) {  weakModel.viewCount = [[data valueForKey:@"viewCount"] integerValue];  } withHttpExceptionBlock:^(NSUInteger code, NSString \*msg) {  } withFailedBlock:^(AFHTTPRequestOperation \*operation, NSError \*error) {    }];  }];    return cell;  }else{  UITableViewCell \*cell = [tableView dequeueReusableCellWithIdentifier:@"otherCell"];  if (!cell) {  cell = [[UITableViewCell alloc]initWithStyle:UITableViewCellStyleDefault reuseIdentifier:@"otherCell"];  }  return cell;  }  }  } |

**4.3.3 首页下子模块页面逻辑**

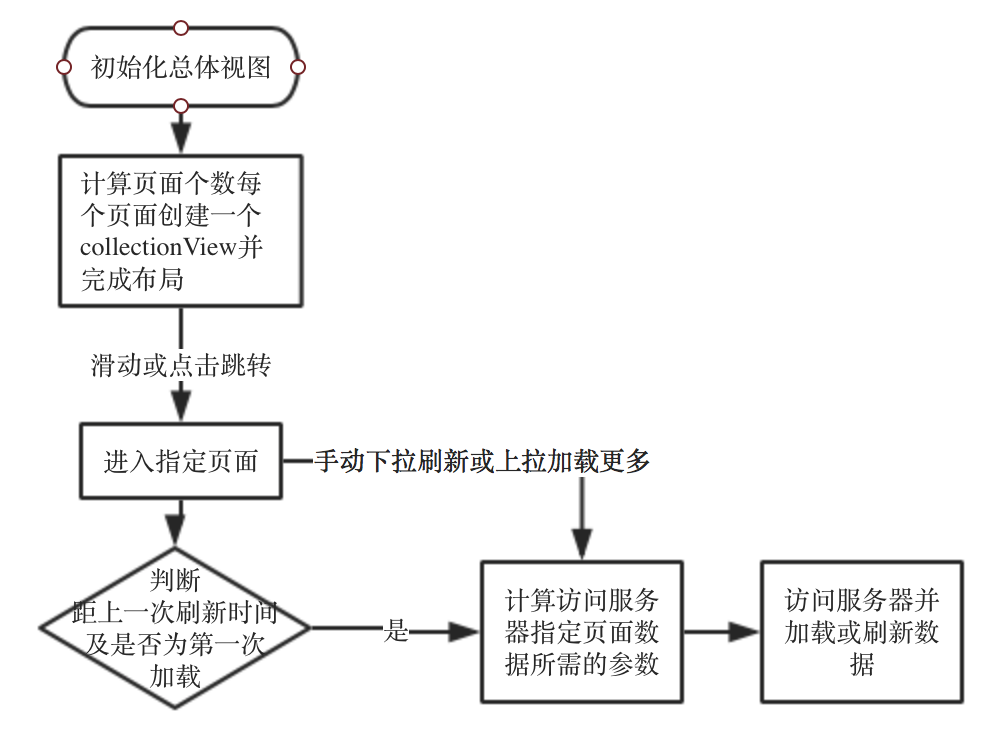
首页下子模块页面个数由获取的服务端数据决定，每个页面以瀑布流的形式展示；支持上拉刷新，下拉加载更多；第一次进入某个页面自动下拉刷新加载数据，之后切换到该页面，在规定时间间隔外才自动刷新。

例如图4-3-3-1服务器返回的属于当前视图的有三个页面：“护肤”，“彩妆”。“时尚”



图4-3-3-1部分导航栏

图4-3-3-2 当前视图其中一个页面效果图

****

|  |
| --- |
| 以下为根据服务端数据创建页面逻辑 |
| **注：**   1. YYNoPsCollectionView只是在系统的UICollectionView基础上封装了一个返回顶部的按钮，其他代理方法等与UICollectionView相同 2. 该视图用initWithFrame创建后要调用setUpWithTabArray:方法才完成创建。但还没加载数据，因为用户打开app后不一定会滑至该页面 |
| - (void)setUpWithTabArray:(NSMutableArray \*)tabArray{  if (!\_tabArray) {  \_tabArray = [NSMutableArray array];  }  \_tabArray = tabArray;  //除去关注，头条，专题3个专题外，其他的才属于本view  if ([tabArray count]>3) {  for (int arrIndex = 2; arrIndex<[tabArray count]-1;arrIndex++) {  YYNoPsCollectionView \*newCollectionView = [self createNewCollectionViewAtIndex:arrIndex-2];  [self addSubview:newCollectionView];  }  }  }  - (YYNoPsCollectionView \*)createNewCollectionViewAtIndex:(NSInteger)index{  UICollectionViewFlowLayout \*layout = [[UICollectionViewFlowLayout alloc] init];  YYNoPsCollectionView \*newCollectionView = [[YYNoPsCollectionView alloc]initWithFrame:CGRectMake(kDeviceWidth\*index, 0, kDeviceWidth, CONTENT\_HEIGHT\_VISIBLE) collectionViewLayout:layout];  [layout setScrollDirection:UICollectionViewScrollDirectionVertical];  newCollectionView.backgroundColor = mochaViewBackgroundColor;  newCollectionView.alwaysBounceVertical = YES;  [newCollectionView registerClass:[YYHomepageOthersCollectionCell class] forCellWithReuseIdentifier:reuseIdentifier];  newCollectionView.delegate = self;  newCollectionView.dataSource = self;  newCollectionView.tag = index + tagValue;    \_\_weak YYHomepageOtherListsView \*wSelf = self;  [newCollectionView addPullToRefreshWithActionHandler:^{  [wSelf loadDataAtIndex:index];  }];  [newCollectionView addInfiniteScrollingWithActionHandler:^{  [wSelf loadMoreDataAtIndex:index];  }];  [newCollectionView.pullToRefreshView setTitle:refreshStateStoppedString forState:SVPullToRefreshStateStopped];  [newCollectionView.pullToRefreshView setTitle:refreshStateTriggeredString forState:SVPullToRefreshStateTriggered];  [newCollectionView.pullToRefreshView setTitle:refreshStateLoadingString forState:SVPullToRefreshStateLoading];  return newCollectionView;  } |

|  |
| --- |
| 每个页面自动刷新，手动刷新来请求加载数据逻辑 |
| 思路：   1. 刷新逻辑主要分为三层：一是供外部调用：刷新第几个页面。此时会判断是否是首次进入或间隔时间，从而选择是否自动刷新，调用刷新函数；二是下拉刷新或上拉加载更多，此方法也只有第几个页面一个参数，该方法主要是计算当前页访问服务器所需参数；第三层便是最后访问服务器的方法函数 2. 包含方法：   -(void)refreshAtIndex:(NSInteger)index  -(void)loadDataAtIndex:(NSInteger)index  -(void)loadMoreDataAtIndex:(NSInteger)index  -(void)requestDataAtIndex:(NSInteger)index withTabID:(NSInteger)tabID  - (void)requestMoreDataAtIndex:(NSInteger)index withTabID:(NSInteger)tabID |
| #pragma mark - public methods  - (void)refreshAtIndex:(NSInteger)index{  NSDate \* refreshTime = [\_refreshTimeDic valueForKey:[NSString stringWithFormat:@"%ld",(long)index]];  NSTimeInterval requestInterval = fabs([refreshTime timeIntervalSinceNow]);  if (0 < requestInterval && requestInterval < 0.5\*60\*60) {  return;  }  YYNoPsCollectionView \* collectionView = [self viewWithTag:tagValue + index];  [collectionView.pullToRefreshView startAnimating];  [self loadDataAtIndex:index];  }  #pragma mark - private methods  - (void)loadDataAtIndex:(NSInteger)index{  if (\_tabArray.count>index + 2) {  YYHomepageTabList \*tab = \_tabArray[index + 2];  NSInteger tabID = [tab.cusID integerValue];  [self requestDataAtIndex:index withTabID:tabID];  }  }  - (void)loadMoreDataAtIndex:(NSInteger)index{  if (\_tabArray.count>index + 2) {  YYHomepageTabList \*tab = \_tabArray[index + 2];  NSInteger tabID = [tab.cusID integerValue];  [self requestMoreDataAtIndex:index withTabID:tabID];  }  }  #pragma mark - request data  - (void)requestDataAtIndex:(NSInteger)index withTabID:(NSInteger)tabID{  \_\_weak typeof(self) weakSelf = self;  NSDictionary \*dic = [NSDictionary dictionaryWithObjectsAndKeys:[NSNumber numberWithInteger:0],@"beginIndex",[NSNumber numberWithInteger:tabID],@"tabID", nil];  [YYHttpUtil httpPostWithAuthInfo:dic withUrl:@"/front/hometab/info" withHttpSuccessBlock:^(id data) {  YYNoPsCollectionView\* collectionView = [weakSelf viewWithTag:tagValue + index];  collectionView.showsInfiniteScrolling = YES;  [collectionView.pullToRefreshView stopAnimating];  if (!weakSelf.modelArray) {  weakSelf.modelArray = [NSMutableArray array];  }  NSMutableArray \*tempArray = [NSMutableArray array];  for (NSDictionary \*tempDic in data){  [tempArray addObject:[YYHomepageOthersCellModel parseData:tempDic]];  }  if ([weakSelf.modelArray count] <= index) {  for (int i = [weakSelf.modelArray count] ; i<index; i++) {  [weakSelf.modelArray addObject:@""];  }  [weakSelf.modelArray addObject:tempArray];  }else{  [weakSelf.modelArray removeObjectAtIndex:index];  [weakSelf.modelArray insertObject:tempArray atIndex:index];  }  [collectionView reloadData];  [\_refreshTimeDic setValue:[NSDate date] forKey:[NSString stringWithFormat:@"%ld",(long)index]];  } withHttpExceptionBlock:^(NSUInteger code, NSString \*msg) {  YYNoPsCollectionView \* collectionView = [weakSelf viewWithTag:tagValue + index];  [YYViewUtil showAlter:collectionView withText:msg withOffset:-40];  [collectionView.pullToRefreshView stopAnimating];  } withFailedBlock:^(AFHTTPRequestOperation \*operation, NSError \*error) {  YYNoPsCollectionView \* collectionView = [weakSelf viewWithTag:tagValue + index];  [YYViewUtil showAlter:collectionView withText:errorText withOffset:-40];  [collectionView.pullToRefreshView stopAnimating];  }];  }  - (void)requestMoreDataAtIndex:(NSInteger)index withTabID:(NSInteger)tabID{  \_\_weak typeof(self) weakSelf = self;  NSMutableArray \*tempArray = [NSMutableArray array];  tempArray = [self.modelArray count]>index ? self.modelArray[index]:nil;  NSDictionary \*dic = [NSDictionary dictionaryWithObjectsAndKeys:[NSNumber numberWithInteger:tempArray.count],@"beginIndex",[NSNumber numberWithInteger:tabID],@"tabID", nil];  [YYHttpUtil httpPostWithAuthInfo:dic withUrl:@"/front/hometab/info" withHttpSuccessBlock:^(id data) {  NSMutableArray \*tempArray = [NSMutableArray array];  for (NSDictionary \*tempDic in data){  [tempArray addObject:[YYHomepageOthersCellModel parseData:tempDic]];  }  if(weakSelf.modelArray.count >index)  [(NSMutableArray \*)weakSelf.modelArray[index] addObjectsFromArray:tempArray] ;  YYNoPsCollectionView \* collectionView = [weakSelf viewWithTag:tagValue + index];  [collectionView reloadData];  [collectionView.infiniteScrollingView stopAnimating];  } withHttpExceptionBlock:^(NSUInteger code, NSString \*msg) {  YYNoPsCollectionView \* collectionView = [weakSelf viewWithTag:tagValue + index];  [YYViewUtil showAlter:collectionView withText:msg withOffset:-40];  [collectionView.infiniteScrollingView stopAnimating];  } withFailedBlock:^(AFHTTPRequestOperation \*operation, NSError \*error) {  YYNoPsCollectionView \* collectionView = [weakSelf viewWithTag:tagValue + index];  [YYViewUtil showAlter:collectionView withText:errorText withOffset:-40];  [collectionView.infiniteScrollingView stopAnimating];  }];  } |

|  |
| --- |
| 以下代码为实现各个collectionView之间的协调逻辑，全部代理方法 |
| 思路：   1. 所有页面的collection共用同一个代理方法，在代理，数据源的方法中判断当前是哪个页面colletionView，从而获取该页面的数据   注：代理方法中部分代码抽出来作为私有方法，未展示 |
| #pragma mark - collectionView delegate  - (NSInteger)collectionView:(YYNoPsCollectionView \*)collectionView numberOfItemsInSection:(NSInteger)section{  int index = collectionView.tag - tagValue;  NSMutableArray \*tempArray = [NSMutableArray array];  tempArray = [self.modelArray count]>index ? self.modelArray[index]:nil;  return tempArray.count;  }  - (UICollectionViewCell \*)collectionView:(YYNoPsCollectionView \*)collectionView cellForItemAtIndexPath:(NSIndexPath \*)indexPath{  int index = collectionView.tag - tagValue;  NSMutableArray \*tempArray = [NSMutableArray array];  tempArray = [self.modelArray count]>index ? self.modelArray[index]:nil;  YYHomepageOthersCellModel \* cellModel = [[YYHomepageOthersCellModel alloc]init];  cellModel = tempArray[indexPath.row];  YYHomepageOthersCollectionCell \*cell = [collectionView dequeueReusableCellWithReuseIdentifier:reuseIdentifier forIndexPath:indexPath];  cell.model = cellModel;  cell.controller = self.controller;  return cell;  }  - (CGSize)collectionView:(YYNoPsCollectionView \*)collectionView layout:(UICollectionViewLayout \*)collectionViewLayout sizeForItemAtIndexPath:(NSIndexPath \*)indexPath{  return CGSizeMake(WIDTH\_CELL, [YYHomepageOthersCollectionCell heightForOthersCollectionCell]);  }  - (CGFloat)collectionView:(YYNoPsCollectionView \*)collectionView layout:(UICollectionViewLayout \*)collectionViewLayout minimumInteritemSpacingForSectionAtIndex:(NSInteger)section  {  return spacing;  }  - (CGFloat)collectionView:(YYNoPsCollectionView \*)collectionView layout:(UICollectionViewLayout \*)collectionViewLayout minimumLineSpacingForSectionAtIndex:(NSInteger)section  {  return spacing;  }  - (UIEdgeInsets)collectionView:(YYNoPsCollectionView \*)collectionView layout:(UICollectionViewLayout \*)collectionViewLayout insetForSectionAtIndex:(NSInteger)section  {  return UIEdgeInsetsMake(spacing, spacing ,spacing, spacing);  }  - (void)collectionView:(YYNoPsCollectionView \*)collectionView didSelectItemAtIndexPath:(NSIndexPath \*)indexPath{  int index = collectionView.tag - tagValue;  NSMutableArray \*tempArray = [NSMutableArray array];  tempArray = [self.modelArray count]>index?self.modelArray[index]:nil;  YYHomepageOthersCellModel \* cellModel = [[YYHomepageOthersCellModel alloc]init];  cellModel = tempArray[indexPath.row];  if (cellModel.dataType == HomeOthers\_Post) {  YYSocialPostViewController \*postVC = [[YYSocialPostViewController alloc] initWithPostId:cellModel.cusID ];  [[self.controller navigationController] pushViewController:postVC animated:YES];  }else if(cellModel.dataType == HomeOthers\_Video){  YYShopVideoDetailViewController \*videoVC = [[YYShopVideoDetailViewController alloc] initVideoId:cellModel.cusID];  videoVC.delegate = self;  [[self.controller navigationController] pushViewController:videoVC animated:YES];  }else{  }  } |

**4.3.4 “种草”引导**

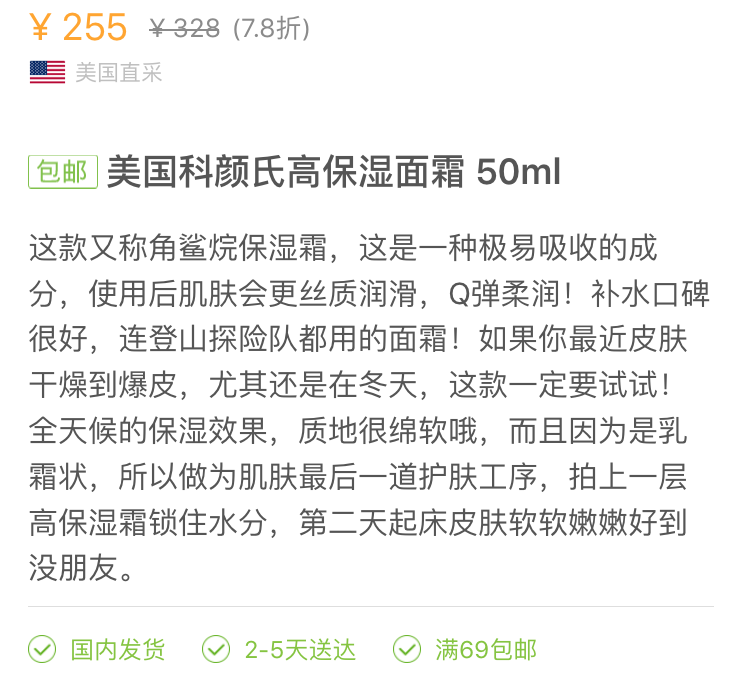
当用户升级到新版本，或第一次安装app,会出现“种草”引导（“种草”有加入心愿单之意，此为新版本新玩法，所以会出现一次引导，之后使用app不会出现）。





|  |
| --- |
| 以下为引导图逻辑及动画实现代码 |
| 注：以下仅为部分引导图初始化，展示，动画，点击事件，其他的实现方法一致 |
| - (UIButton \*)firstGuideButton{  if (!\_firstGuideButton) {  \_firstGuideButton = [[UIButton alloc]init];  \_firstGuideButton.showsTouchWhenHighlighted = NO;  \_firstGuideButton.adjustsImageWhenHighlighted = NO;  [\_firstGuideButton setBackgroundImage:[UIImage imageNamed:@"guidance\_1"] forState:UIControlStateNormal];  [\_firstGuideButton addTarget:self action:@selector(clickFirstButton) forControlEvents:UIControlEventTouchUpInside];  }  return \_firstGuideButton;  }  -(void)clickThirdButton{  [self addSubview:self.fourthGuideButton];  \_thirdGuideButton.userInteractionEnabled = NO;  [\_fourthGuideButton autoPinEdgesToSuperviewEdgesWithInsets:UIEdgeInsetsZero];  UIImageView \*fourthCoverView = [[UIImageView alloc]initWithImage:[UIImage imageNamed:@"guidance\_4\_mask"]];  [\_fourthGuideButton addSubview:fourthCoverView];  fourthCoverView.alpha = 0;  [fourthCoverView autoPinEdgesToSuperviewEdgesWithInsets:UIEdgeInsetsZero];  [UIView animateWithDuration: 0.5 animations: ^{  fourthCoverView.alpha = 1;  } completion: nil];    UIImageView \* infoIcon = [[UIImageView alloc]init];  infoIcon.image = [UIImage imageNamed:@"guidance\_4\_dialog\_box" ];  [\_fourthGuideButton addSubview:infoIcon];  infoIcon.tag = 1112;  //根据图标各比例算的距离  [infoIcon autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:kDeviceHeight\*850/1334 - 35];  [infoIcon autoPinEdgeToSuperviewEdge:ALEdgeRight withInset:kDeviceWidth\*10/75 + 128];  [infoIcon autoSetDimensionsToSize:CGSizeMake(136, 90)];  CGPoint InfoCenter = infoIcon.center;  CGPoint InfoStartCenter = InfoCenter;  InfoStartCenter.y+= kDeviceHeight;  infoIcon.center = InfoStartCenter;    UIImageView \* icon = [[UIImageView alloc]init];  icon.image = [UIImage imageNamed:@"guidance\_4\_chayao" ];  [\_fourthGuideButton addSubview:icon];  icon.tag = 1111;  //根据图标各比例算的距离  [icon autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:kDeviceHeight\*850/1334];  [icon autoPinEdgeToSuperviewEdge:ALEdgeRight withInset:kDeviceWidth\*10/75];  [icon autoSetDimensionsToSize:CGSizeMake(128, 145)];  CGPoint center = icon.center;  CGPoint startCenter = center;  startCenter.y += kDeviceHeight;  icon.center = startCenter;    [UIView animateWithDuration: 1.5 delay: 0.5 usingSpringWithDamping: 0.9 initialSpringVelocity: 0 options: UIViewAnimationOptionCurveLinear animations: ^{  icon.center = center;  infoIcon.center = InfoCenter;  } completion: ^(BOOL finished){  \_fourthGuideButton.userInteractionEnabled = YES;  }];  dispatch\_after(dispatch\_time(DISPATCH\_TIME\_NOW, (int64\_t)(0.2 \* NSEC\_PER\_SEC)), dispatch\_get\_main\_queue(), ^{  icon.animationImages = [NSArray arrayWithObjects:  [UIImage imageNamed:@"guidance\_4\_chayao"],  [UIImage imageNamed:@"guidance\_4\_chayao1"],  [UIImage imageNamed:@"guidance\_4\_chayao2"],  [UIImage imageNamed:@"guidance\_4\_chayao3"],  [UIImage imageNamed:@"guidance\_4\_chayao4"],  [UIImage imageNamed:@"guidance\_4\_chayao5"], nil];  icon.animationDuration = 2;  [icon startAnimating];  });    }  -(void)clickFourthButton{  \_fourthGuideButton.userInteractionEnabled = NO;  UIImageView \* chayaoIcon = [\_fourthGuideButton viewWithTag:1111];  UIImageView \* infoIcon = [\_fourthGuideButton viewWithTag:1112];  [UIView animateWithDuration:1 delay:0 options:UIViewAnimationOptionCurveEaseInOut animations:^{  chayaoIcon.alpha = 0;  infoIcon.alpha = 0;  } completion:^(BOOL finished) {  [chayaoIcon removeFromSuperview];  [infoIcon removeFromSuperview];  }];  UIImageView \* icon = [[UIImageView alloc]init];  icon.image = [UIImage imageNamed:@"guidance\_4\_money" ];  [\_fourthGuideButton addSubview:icon];  [icon autoAlignAxisToSuperviewAxis:ALAxisHorizontal];  [icon autoAlignAxisToSuperviewAxis:ALAxisVertical];  [icon autoSetDimension:ALDimensionWidth toSize:kDeviceWidth/4\*3];  CGPoint center = icon.center;  CGPoint startCenter = center;  startCenter.y += -260-113;  icon.center = startCenter;    UIButton \*cancelButton = [UIButton buttonWithType:UIButtonTypeCustom];  cancelButton.backgroundColor = [UIColor clearColor];  cancelButton.userInteractionEnabled = NO;  [cancelButton addTarget:self action:@selector(removeGuideView) forControlEvents:UIControlEventTouchUpInside];  [self addSubview:cancelButton];  [cancelButton autoPinEdgesToSuperviewEdgesWithInsets:UIEdgeInsetsZero];  [UIView animateWithDuration: 1.5 delay:0.5 usingSpringWithDamping: 0.9 initialSpringVelocity: 0 options: UIViewAnimationOptionCurveLinear|UIViewAnimationOptionTransitionCurlUp animations: ^{  icon.center = center;  } completion: ^(BOOL finished){  cancelButton.userInteractionEnabled = YES;  }];    }  - (void)removeGuideView{  [UIView animateWithDuration:1 delay:0 options:UIViewAnimationOptionCurveEaseInOut animations:^{  self.alpha = 0;  } completion:^(BOOL finished) {  [self removeFromSuperview];  }];  } |

4.3.5 视图精确布局



|  |
| --- |
| 文字等之间精确布局 |
| 说明：有时产品需求上会要求文字图片模块之间严格的间距，并且某些小内容会动态显示与否，因此必须考虑各种组合之间的相互布局约束 |
| - (void)reLayoutWithData:(YYGoodsDetail \*)goods yyGoodType:(YYGoodsDetailVCType)type  {  NSString \*goodsName = nil;//商品名  NSString \*salePrice = nil;//销售价  NSString \*originPrice = nil;//原始价格  NSString \*discount = nil;//折扣  NSString \*groupBuyPrice = nil;//拼团价  groupBuyPrice = goods.groupBuy? [NSString stringWithFormat:@"拼团价 ¥ %@", goods.groupBuy.price] :@"暂定";  BOOL haveStock = NO;//是否有库存  //普通商品赋值  if (type == YYGoodsDetailVCNormal) {  goodsName = goods.name;  YYPropValue \*price;  for (YYPropValue \*subValue in goods.propPrice.propValueList) {  if (1 == subValue.isDefault.integerValue) {  price = subValue;//商品本身若无价格，则显示默认价格  }  }  salePrice =price.salePrice? [NSString stringWithFormat:@"¥ %@", price.salePrice ] :@"";  originPrice = (price.originPrice.floatValue>0)? [NSString stringWithFormat:@"¥ %@", price.originPrice ]:@"";  if (price.discount.floatValue > 0 && price.discount.floatValue<10.0) {  discount = [NSString stringWithFormat:@"(%.1f折)",price.discount.floatValue] ;}else{  discount = @"";  }  haveStock = [goods.totalStockCount integerValue] >0 ? YES : NO;  }else if(type == YYGoodsDetailVCSecondsKill){//秒杀商品赋值  goodsName = goods.seckill.name;  salePrice =goods.seckill.salePrice? [NSString stringWithFormat:@"¥ %@", goods.seckill.salePrice ] :@"";  originPrice = (goods.seckill.originPrice.floatValue>0)? [NSString stringWithFormat:@"¥ %@", goods.seckill.originPrice ]:@"";  if (goods.seckill.discount.floatValue>0 && goods.seckill.discount.floatValue <10) {  discount = [NSString stringWithFormat:@"(%.1f折)",goods.seckill.discount.floatValue] ;  }else{ discount = @"" ; }  haveStock = [goods.seckill.currStock integerValue] >0 ? YES : NO;  }  //心愿单图标布局  if (haveStock) {  \_btnWant.hidden = YES;  }else{  \_btnWant.hidden = NO;  if (goods.iWanted) {  \_btnWant.selected = YES;  }else{  \_btnWant.selected = NO;  }  [\_btnWant autoSetDimensionsToSize:CGSizeMake(78, 26)];  [\_btnWant autoPinEdgeToSuperviewEdge:ALEdgeRight withInset:rightMargin];  [\_btnWant autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:37];  }  //价格布局  \_labPrice.text = salePrice;  [\_labPrice autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:topMargin];  [\_labPrice autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];    NSDictionary \*attrs = @{NSStrikethroughStyleAttributeName:@(NSUnderlineStyleSingle|NSUnderlinePatternSolid),NSStrikethroughColorAttributeName:[UIColor YYcolorWithHexInt:0x8b8b8b]};  NSAttributedString \*attOldPrice = [[NSAttributedString alloc]initWithString:originPrice attributes:attrs];  \_labOldPrice.attributedText = attOldPrice;  [\_labOldPrice autoPinEdge:ALEdgeLeft toEdge:ALEdgeRight ofView:\_labPrice withOffset:horizontalSpacing];  [\_labOldPrice autoPinEdge:ALEdgeBottom toEdge:ALEdgeBottom ofView:\_labPrice withOffset:-3.0];    \_discountLabel.text = discount;  [\_discountLabel autoPinEdge:ALEdgeLeft toEdge:ALEdgeRight ofView:\_labOldPrice withOffset:horizontalSpacingBetweenOldPriceAndDiscount];  [\_discountLabel autoPinEdge:ALEdgeBottom toEdge:ALEdgeBottom ofView:\_labOldPrice];    //原产地信息布局&赠送信息布局  CGFloat flagIconHeight = [@"占位符" sizeWithFont:SHIPPING\_FONT constrainedToSize:CGSizeMake(200, MAXFLOAT)].height;  CGFloat flagIconWidth = flagIconHeight/12\*18;    NSString \* giftInfoString = (![goods.gift.content isEqualToString: @""] &&goods.gift!=nil)?[@"赠品:"stringByAppendingString :goods.gift.content]:nil;//避免是空字符串的情况  if (!giftInfoString) {  \_giftInfoButton.hidden = YES;  }else{  \_giftInfoButton.hidden = NO;  }  [\_giftInfoButton setAttributedTitle:[YYSocialHighlightString formatGiftInfoButtonWithNSString:giftInfoString] forState:UIControlStateNormal];  [\_shippingTypeIcon autoSetDimensionsToSize:CGSizeMake(flagIconWidth, flagIconHeight)];  [\_shippingTypeIcon autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_labPrice withOffset:verticalSpacingBetweenPriceAndOrigin];  [\_shippingTypeIcon autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];  if (goods.goodsOrigin) {  [\_shippingTypeIcon setImageURLForUpyunWithURL: goods.goodsOrigin.picUrl width:flagIconWidth height:flagIconHeight placeHolderImage:nil];  \_shippingTypeIcon.hidden = NO;  \_shippingTypeLabel.text = goods.goodsOrigin.content;  [\_shippingTypeLabel autoPinEdge:ALEdgeLeft toEdge:ALEdgeRight ofView:\_shippingTypeIcon withOffset:4];  [\_shippingTypeLabel autoPinEdge:ALEdgeBottom toEdge:ALEdgeBottom ofView:\_shippingTypeIcon ];  [\_shippingTypeLabel setContentHuggingPriority:UILayoutPriorityDefaultHigh forAxis:UILayoutConstraintAxisHorizontal];    [\_giftInfoButton autoPinEdge:ALEdgeLeft toEdge:ALEdgeRight ofView:\_shippingTypeLabel withOffset:horizontalSpacingBetweenOriginAndGiftInfo];  [\_giftInfoButton autoAlignAxis:ALAxisHorizontal toSameAxisOfView:\_shippingTypeIcon];  }else{  \_shippingTypeIcon.hidden = YES;  [\_giftInfoButton autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];  [\_giftInfoButton autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_labPrice withOffset:verticalSpacingBetweenPriceAndOrigin];  }  [\_giftInfoButton autoPinEdgeToSuperviewEdge:ALEdgeRight];    //拼团说明view  if (!goods.goodsOrigin && !giftInfoString) {  [\_groupBuyStatus autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_labPrice withOffset:verticalSpacingAboutGroupBuy];  }else{  [\_groupBuyStatus autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_shippingTypeIcon withOffset:verticalSpacingAboutGroupBuy];  }  [\_groupBuyStatus autoPinEdgeToSuperviewEdge:ALEdgeLeft];  [\_groupBuyStatus autoSetDimension:ALDimensionWidth toSize:kDeviceWidth];  //拼团相关  if(goods.groupBuy)  {  \_labPrice.hidden = YES;\_labOldPrice.hidden = YES; \_discountLabel.hidden = YES;  \_groupPriceLabel.hidden = NO;\_notGroupPriceLabel.hidden = NO;    \_notGroupPriceLabel.text = salePrice;  [\_notGroupPriceLabel autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];  [\_notGroupPriceLabel autoPinEdgeToSuperviewEdge:ALEdgeTop withInset:topMargin];    \_groupPriceLabel.text = groupBuyPrice;  [\_groupPriceLabel autoAlignAxis:ALAxisHorizontal toSameAxisOfView:\_notGroupPriceLabel];  [\_groupPriceLabel autoPinEdge:ALEdgeLeft toEdge:ALEdgeRight ofView:\_notGroupPriceLabel withOffset:horizontalSpacing];    [\_groupBuyStatus autoSetDimension:ALDimensionHeight toSize: [YYGroupBuyStatusView viewHeightHasDescription:YES]];    [\_labTitle autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_groupBuyStatus withOffset:verticalSpacingAboutGroupBuy];  }else{  \_labPrice.hidden = NO;\_labOldPrice.hidden = NO; \_discountLabel.hidden = NO;  \_groupPriceLabel.hidden = YES;\_notGroupPriceLabel.hidden = YES;    \_groupBuyStatus.hidden = YES;  if (!goods.goodsOrigin && !giftInfoString) {  [\_groupBuyStatus autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_labPrice];  }else{  [\_groupBuyStatus autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_shippingTypeIcon];  }  [\_groupBuyStatus autoSetDimension:ALDimensionHeight toSize:0];    [\_labTitle autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_groupBuyStatus withOffset:verticalSpacingBetweenTitleAndOrigin];  }  //商品名布局  CGFloat saleInfoHeight = [@"预售包邮占位符" sizeWithFont:TITLE\_FONT constrainedToSize:CGSizeMake(200, MAXFLOAT)].height;  if (goods.saleInfo) {  YYSaleInfo \*tempSaleInfo = goods.saleInfo[0];  [\_infoBeforeTitleImageView setImageURLForUpyunWithURL:tempSaleInfo.picUrl width:35 height:saleInfoHeight placeHolderImage:nil];  [\_infoBeforeTitleImageView autoPinEdge:ALEdgeTop toEdge:ALEdgeTop ofView:\_labTitle];  [\_infoBeforeTitleImageView autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];  [\_infoBeforeTitleImageView autoSetDimensionsToSize:CGSizeMake(35, saleInfoHeight)];  \_labTitle.text = [self appendBlankSpaceBeforeNSString:goodsName ];  }else{  \_labTitle.text = [NSString stringWithFormat:@"%@",goodsName];  }  [\_labTitle autoPinEdgeToSuperviewEdge:ALEdgeLeft withInset:leftMargin];  [\_labTitle autoPinEdgeToSuperviewEdge:ALEdgeRight withInset:leftMargin];    //商品介绍view  if (!\_introductionView) {  \_introductionView = [[YYDetailedIntroductionView alloc]initWithGoodModel:goods];  [self addSubview:\_introductionView];  }  [\_introductionView autoPinEdgeToSuperviewEdge:ALEdgeLeft];  [\_introductionView autoPinEdgeToSuperviewEdge:ALEdgeRight];  [\_introductionView autoPinEdge:ALEdgeTop toEdge:ALEdgeBottom ofView:\_labTitle withOffset:verticalSpacingBetweenTitleAndDetail];  [\_introductionView autoSetDimension:ALDimensionHeight toSize:[\_introductionView heightForDetailedIntroductionView]];  } |