**约定URI请求格式**

html页面与app通过指定的特殊格式的接口进行交互请求。对应请求模板如下：

goodsapp://nav/openLinkInNewWebView?p=XXX

其中

*goodsapp*: scheme头，用来与app进行通讯标志，标志本次请求为页面与app交互请求

*nav* ：host部分，该部分用于app进行定位对应的功能模块Controller

*openLinkInNewWebView* ： path部分，该部分用于app定位对应功能模块下的具体方法

*p=XXX* ： query部分，该部分为app具体方法所对应的接收参数，XXX为json格式字符串。app在接收到对应参数后，可以进行相对应的方法逻辑处理

**后台 mgr/backend**

1.修改目前的活动模板标识符。活动位广告现在的活动类型固定为三种模板，模板名分别是：wholes\_sale，clear\_out，new\_arrival。这三种模板对应现在web端的Goods Wow，Clearance以及New Arrivals。后台维护的时候要选择需要的对应模板。APP都有内置对应的功能模板，广告用Native方式呈现。

2.商品详情位广告直接定位到APP中对应的内置商品详情页，广告用native方式呈现。后台维护的时候要填写需要推广的productId。

3.目录列表位广告直接定位到APP中对应的内置目录列表页，广告用native方式呈现。后台维护的时候要填写需要推广的categoryId。

1. 新增一种广告类型。新增的广告类型为三方html页面位广告，需要APP调用WebView展示外部链接广告，同时相应广告页面对应与APP之间的操作。后台维护的时候要填写需要推广的广告页面地址。
2. 现在后台手机广告添加包括四种类型，分别是活动位广告、商品详情位广告、目录列表位广告以及第三方html页面位广告。

**前台 goodscart**

1.引入第三方js文件：LDJSBridge.js。用于WebView和APP Native之间进行交互的枢纽，其本质是通过嵌入iframe的方式，进行约定好对应URI scheme的请求，从而使app能够相应对应的操作

2.引入自己编写的js文件：jsBridgeFunction.js。封装LDJSBridge.js的方法，用于APP内嵌的html广告页进行直接调取使用。注意一点，如果需要出html的广告页，同时应用于APP内嵌的WebView以及手机端的浏览器进行访问，则这里需要出两套不同的对应html页面，一套是专门给浏览器使用的，可以进行正常的链接跳转，同时在展示出对应的页面之后，给出跳转APP的提示信息；一套是专门给APP内置的WebView使用的，内部链接全部用jsBridgeFunction.js中定义好的方法代替。点击的时候不进行链接跳转，而是直接调起APP的响应，由APP决定跳转到那个对应界面

3.设置html5页中的埋点信息，添加对应的请求操作。目的是在用户访问手机版的Goods商城的时候，如果安装的有Goods APP，则会自动调起APP，跳转到对应的界面。目前已经添加埋点的模块有以下几个：html5的商品详情页，html5的目录列表页以及html5的三个活动页（html5首页看情况，看要不要也加上自动跳转APP的方法）

4.设置页面顶部banner引导条（待定）。所有html5页面添加页顶的banner条，引导用户下载（引导方式可能为送券，或者活动促销），或者跳转APP进行访问。banner条点击逻辑如下：点击按钮之后，跳转到专门的APP下载引导页。在引导页包含下载点击按钮和添加的埋点，当用户本身安装的有APP的时候，会提示用户自动跳转APP；当用户没有安装APP的时候，则需要用户点击按钮进行下载链接跳转（下载链接采用直接调用系统商城的方法，调用不成功的话，则进行APP下载链接跳转）

**mobile**

1.修改首页的广告栏接口返回参数。去掉linktype等多余字段，只保留url以及redirectUrl两个参数，url为广告图片的地址，redirectUrl为协议好URI scheme的跳转地址。这么做的目的是：一、保持广告请求参数的整体简洁；二、保证接口端、APP WebView端以及浏览器端调用规则的一致；三、确保以后假如APP首页变成html5模式了，该接口仍然能正常使用