# 我的商城主线笔记

### 启动页

#### 去标题栏

**public class** WelcomeActivity **extends** Activity

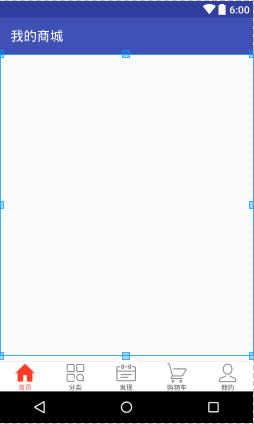
#### 延时3s进入主页面中的延时实现

**private** Handler **handler** = **new** Handler();

**handler**.postDelayed(**new** Runnable() **{** @Override  
 **public void** run() **{** // 进到主页面  
 finish(); *// 关闭本页面* **}  
}**, 3000);

### MainActivity主页

#### 主页布局（整体：线性垂直布局）



RadioGroup水平布局（带背景）

帧布局

##### RadioButton代码

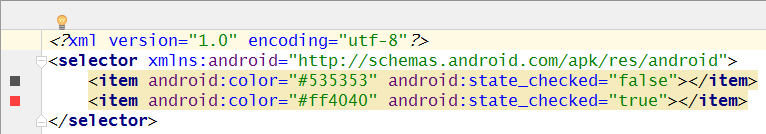
// 首页RadioButton

<**RadioButton** android:id="@+id/rb\_home"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/home\_button\_selector"  
 android:text="@string/rb\_home"  
 android:checked="true"/>

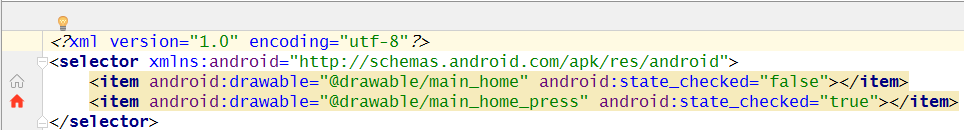
##### RadioButton样式

<**style** name="MainButtonStyle">  
 *<!-- Customize your theme here. -->* <**item** name="android:layout\_width">0dp</**item**>  
 <**item** name="android:layout\_height">wrap\_content</**item**>  
 <**item** name="android:layout\_weight">1</**item**>  
 <**item** name="android:button">@null</**item**>  
 *<!-- <item name="android:drawablePadding">3dp</item>-->* <**item** name="android:textColor">@drawable/bottom\_button\_text\_selector</**item**> // 文本选择器  
 <**item** name="android:textSize">10sp</**item**>  
 <**item** name="android:gravity">center</**item**>  
</**style**>

bottom\_button\_text\_selector文本选择器

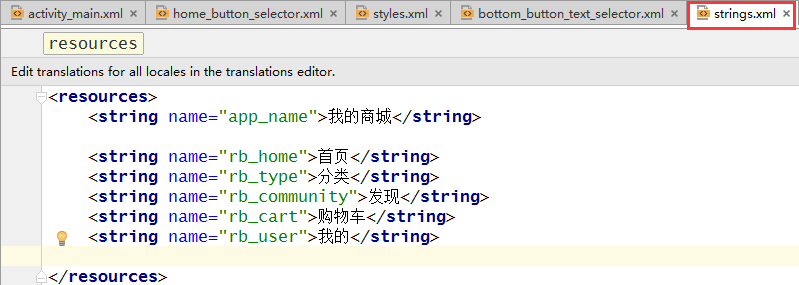


##### RadioButton顶部按钮选择器



##### RadioButton名称管理

android:text="@string/rb\_home"



##### 完整代码（注意中间的RadioButton）

*<?*xml version="1.0" encoding="utf-8"*?>*<**LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="com.myemcu.myshop.app.MainActivity">  
  
 <**FrameLayout** android:id="@+id/frame\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="0dp"  
 android:layout\_weight="1"/>  
  
 <**RadioGroup** android:id="@+id/rg\_main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/home\_bottom\_parent\_bg"  
 android:orientation="horizontal">  
  
 <**RadioButton** android:id="@+id/rb\_home"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/home\_button\_selector"  
 android:text="@string/rb\_home"  
 android:checked="true"/>  
  
 <**RadioButton** android:id="@+id/rb\_type"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/type\_button\_selector"  
 android:text="@string/rb\_type"/>  
  
 <**RadioButton** android:id="@+id/rb\_community"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/community\_button\_selector"  
 android:paddingTop="10dp"  
 android:text="@string/rb\_community"/>  
  
 <**RadioButton** android:id="@+id/rb\_cart"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/cart\_button\_selector"  
 android:text="@string/rb\_cart"/>  
  
 <**RadioButton** android:id="@+id/rb\_user"  
 style="@style/MainButtonStyle"  
 android:drawableTop="@drawable/user\_button\_selector"  
 android:text="@string/rb\_user"/>  
  
 </**RadioGroup**>  
  
</**LinearLayout**>

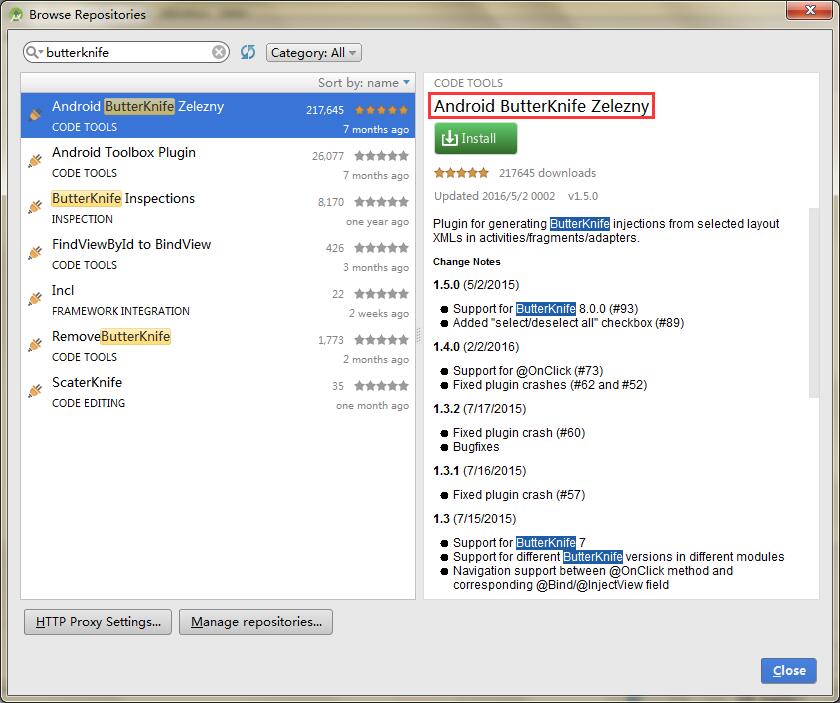
#### 使用Butterknife实例化主页控件

##### 添加依赖（须联网）

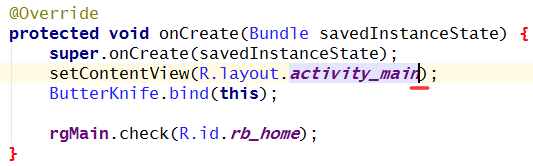
compile 'com.jakewharton:butterknife:7.0.1' // 稳定版，8.4.0的不稳定

##### 下载插件

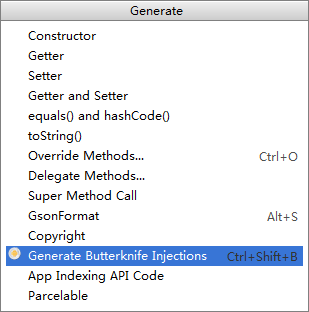
如下图所示：



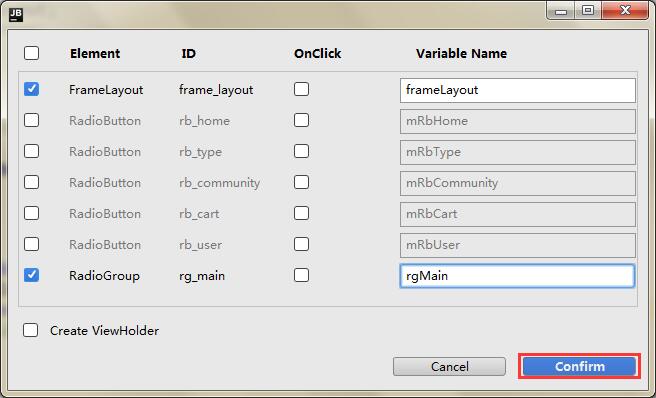
##### 设置光标位置



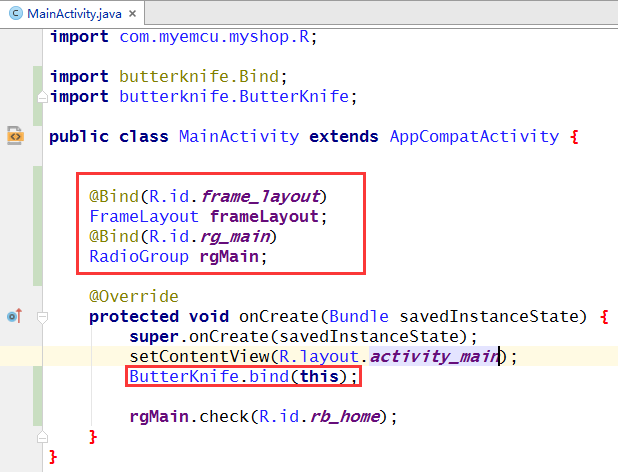
##### Alt+Insert



##### 选择需要被实例化的控件



##### 实现效果



### 系统框架——软件分包

#### 功能分包

* app——Activity
* base——公共基类
* home
* type
* community
* cart
* user

——上面5个，子Fragment。

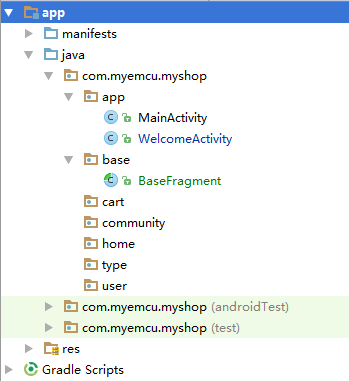
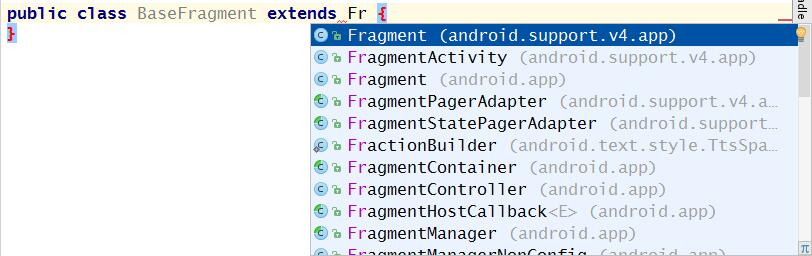


图 1.3.1 软件分包

#### 5个Fragment页面的公共Fragment

##### 继承时用V4包



##### BaseFragment

**public abstract class** BaseFragment **extends** Fragment **{  
  
 protected** Context **context**; *// (public全公开；privite不公开，仅自己用；protected公开到熟人)* @Override **public void onCreate**(@Nullable Bundle savedInstanceState) **{** *// 该类被系统创建时回调* **super**.onCreate(savedInstanceState);  
 **context**=getActivity();  
 **}** @Nullable  
 @Override  
 **public** View **onCreateView**(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle

savedInstanceState) **{  
 return** initView();  
 **}**

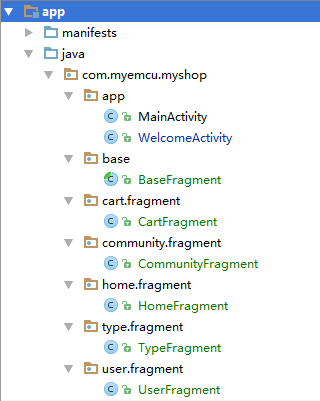
**public abstract** View initView(); *// 由继承它的子Fragment强制实现，当然基类也必须是抽象的* @Override *// 当Activity被创建的时候，回调该方法* **public void onActivityCreated**(@Nullable Bundle savedInstanceState) **{  
 super**.onActivityCreated(savedInstanceState);  
 initData();  
 **}****public void** initData() **{** *// 当子Fragment需要联网请求数据的时候，可重写该方法* **}  
}**

### 简单构建5个Fragment页面内容

#### 以如下HomeFragment为模板

*// 首页Fragment***public class** HomeFragment **extends** BaseFragment**{  
  
 private** TextView **textView**;  
  
 @Override  
 **public** View **initView**() **{  
 textView**=**new** TextView(**context**);  
  
 Log.e("TAG","首页Fragment-UI-已初始化...");  
  
 **textView**.setGravity(Gravity.***CENTER***);  
 **textView**.setTextSize(25);  
 **textView**.setTextColor(Color.***BLUE***);  
 **return textView**;  
 **}** @Override  
 **public void initData**() **{  
 super**.initData();  
  
 Log.e("TAG","首页Fragment-数据-已初始化...");  
  
 **textView**.setText("我是首页...");  
 **}  
}**

#### 完善其它Fragment（卡卡西模式）



### 在MainActivity中，监听切换5个Fragment（UI框架完）

#### 添加Fragment

**private** ArrayList<BaseFragment> **fragments**;

**fragments** = **new** ArrayList<>(); *// 新建集合对象***fragments**.add(**new** HomeFragment()); *// 添加 首页 Fragment***fragments**.add(**new** TypeFragment()); *// 添加 分类 Fragment***fragments**.add(**new** CommunityFragment()); *// 添加 社区 Fragment***fragments**.add(**new** CartFragment()); *// 添加 购物车 Fragment***fragments**.add(**new** UserFragment()); *// 添加 用户 Fragment*

#### 监听RadioGroup

**rgMain**.**setOnCheckedChangeListener**(**new** RadioGroup.OnCheckedChangeListener() **{** @Override  
 **public void** onCheckedChanged(RadioGroup group, **int** checkIid) **{** **}  
}**);

#### 取得当前位置的Fragment

**private** BaseFragment getFragment(**int** postion) **{**

**}**

#### 切换到需要显示的Fragment

**private void** switchFragment(Fragment preFragment, BaseFragment currentFragment) **{  
  
}**

#### 完整代码

**public class** MainActivity **extends FragmentActivity** **{** // 1 去标题栏 2 支持getSupportFragmentManager() **private** ArrayList<BaseFragment> **fragments**; **private int postion** = 0;  
  
 **private** Fragment **tempFragment**; *// 临时Fragment(缓存Fragment)* **private** BaseFragment **currentFragment**; *// 当前Fragment* @Bind(R.id.***frame\_layout***)  
 FrameLayout **frameLayout**;  
 @Bind(R.id.***rg\_main***)  
 RadioGroup **rgMain**;  
  
 @Override  
 **protected void onCreate**(Bundle savedInstanceState) **{  
 super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 ButterKnife.bind(**this**); *// 自动生成* initFragment(); *// 初始化所有Fragment* initListener(); *// 初始化页面控件监听* **}** *// 5个Fragment页面(按顺序添加)* **private void** initFragment() **{  
 fragments** = **new** ArrayList<>(); *// 新建集合对象* **fragments**.add(**new** HomeFragment()); *// 添加 首页 Fragment* **fragments**.add(**new** TypeFragment()); *// 添加 分类 Fragment* **fragments**.add(**new** CommunityFragment()); *// 添加 社区 Fragment* **fragments**.add(**new** CartFragment()); *// 添加 购物车 Fragment* **fragments**.add(**new** UserFragment()); *// 添加 用户 Fragment* **}  
  
 private void** initListener() **{** *// 首页中的各种监听* **rgMain**.setOnCheckedChangeListener(**new** RadioGroup.OnCheckedChangeListener() **{** @Override  
 **public void** onCheckedChanged(RadioGroup group, **int** checkIid) **{  
 switch** (checkIid) **{  
 case** R.id.***rb\_home***: **postion** = 0;  
 **break**;  
  
 **case** R.id.***rb\_type***: **postion** = 1;  
 **break**;  
  
 **case** R.id.***rb\_community***: **postion** = 2;  
 **break**;  
  
 **case** R.id.***rb\_cart***: **postion** = 3;  
 **break**;  
  
 **case** R.id.***rb\_user***: **postion** = 4;  
 **break**;  
  
 **default**: **postion** = 0; **break**;  
 **}** *// 根据位置取当前Fragment* **currentFragment** = getFragment(**postion**);  
 *// 切换Fragment* switchFragment(**tempFragment**, **currentFragment**); *// 临时Fragment，当前取出的Fragment* **}  
 }**);  
  
 **rgMain**.check(R.id.***rb\_home***); *// 默认选中首页(用8.4.0的Butterknife这里要崩)* **}** *// 根据位置，得到相应Fragment* **private** BaseFragment getFragment(**int** postion) **{  
  
 if** (**fragments** != **null** && **fragments**.size()>0) **{** BaseFragment baseFragment = **fragments**.get(postion);  
 **return** baseFragment;  
 **}else {  
 return null**;  
 **}  
 }** *// 根据上次preFragment与当前的Fragment进行切换* **private void** switchFragment(Fragment preFragment, BaseFragment currentFragment) **{**

**if** (**tempFragment** != currentFragment) **{** *// Fragment一样，就不需要处理了。* **tempFragment** = currentFragment;

**if** (currentFragment != **null**) **{**

FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();  
 **if** (!currentFragment.isAdded()) **{  
 if** (preFragment != **null**) **{** transaction.hide(preFragment);  
 **}** transaction.add(R.id.***frame\_layout***,currentFragment).commit();  
 **}else {  
 if** (preFragment != **null**) **{** transaction.hide(preFragment);  
 **}** transaction.show(currentFragment).commit();  
 **}  
 }** *// 验证同一个地方只加载一次Fragment  
 // Toast.makeText(MainActivity.this,"我进来了",Toast.LENGTH\_SHORT).show();* **}  
 }  
}**

#### 实现效果



### 实现首页HomeFragment布局

#### 布局方案（整体相对布局）

* Titlebar
* RecyclerView
* ImageView

#### Titlebar

##### 实现效果



##### 实现代码

*<?*xml version="1.0" encoding="utf-8"*?>*<**LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:orientation="horizontal"  
 android:background="#ed3f3f"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
  
 <**TextView** android:id="@+id/tv\_search\_home"  
 android:layout\_width="0dp"  
 android:layout\_weight="1"  
 android:layout\_height="35dp"  
 android:layout\_marginLeft="10dp"  
 android:text="输入搜索信息"  
 android:textSize="13sp"  
 android:drawablePadding="10dp"  
 android:padding="5dp"  
 android:background="@drawable/search\_home\_shape"  
 android:drawableLeft="@drawable/home\_search\_icon" // 放大镜图标  
 android:layout\_gravity="center"  
 android:gravity="center\_vertical"  
 tools:ignore="HardcodedText,RtlHardcoded"/>  
  
 <**TextView** android:id="@+id/tv\_message\_home"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="10dp"  
 android:text="消息"  
 android:textColor="#fff"  
 android:drawableTop="@drawable/new\_message\_icon" // 信封图标  
 tools:ignore="HardcodedText"/>  
  
</**LinearLayout**>

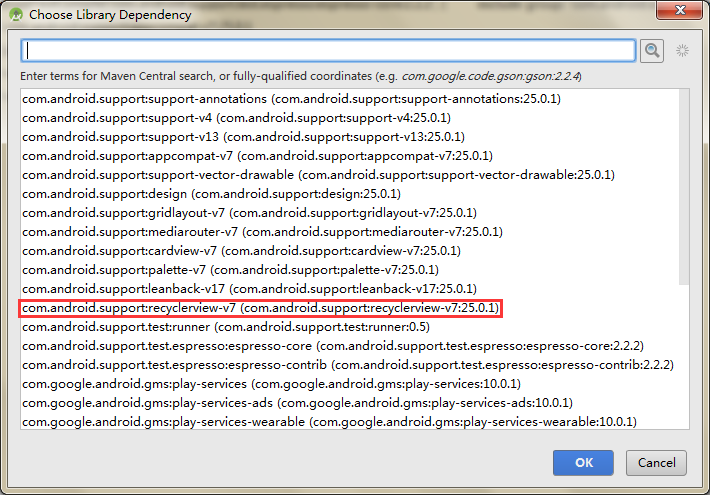
其中，前一个TextView的背景代码如下

*<?*xml version="1.0" encoding="utf-8"*?>*<**shape** xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="rectangle">  
 <**solid** android:color="#fff" />  
 <**corners** android:radius="20dp" />  
</**shape**>

#### RecyclerView

<**android.support.v7.widget.RecyclerView** android:id="@+id/rv\_home"  
 android:layout\_below="@+id/titlebar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>

说明：必须先添加依赖（版本要与com.android.support:appcompat-v7:25.0.1一致）



#### ImageView

实现代码

<**ImageView** android:id="@+id/ib\_top"  
 android:layout\_width="40dp"  
 android:layout\_height="40dp"  
 android:background="@drawable/top\_btn"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginBottom="20dp"  
 android:layout\_marginRight="20dp"  
 android:visibility="visible"  
 tools:ignore="ContentDescription,RtlHardcoded"/>

说明：图片必须是透明背景的.png格式，本图大小256x256

#### 整体实现

##### 实现代码

*<?*xml version="1.0" encoding="utf-8"*?>*<**RelativeLayout** xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <**include** android:id="@+id/titlebar"  
 layout="@layout/titlebar"/>  
 <**android.support.v7.widget.RecyclerView** android:id="@+id/rv\_home"  
 android:layout\_below="@+id/titlebar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>  
  
 <**ImageView** android:id="@+id/ib\_top"  
 android:layout\_width="40dp"  
 android:layout\_height="40dp"  
 android:background="@drawable/top\_btn"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginBottom="20dp"  
 android:layout\_marginRight="20dp"  
 android:visibility="visible"  
 tools:ignore="ContentDescription,RtlHardcoded"/>  
  
</**RelativeLayout**>

##### 实现效果



### 初始化首页HomeFragment布局

#### 实例化RecyclerView、ImageView、TextView（两个）

#### 监听ImageView、TextView（两个）

#### 关键代码

**recy\_home** = (RecyclerView) view.findViewById(R.id.***recy\_home***);  
**iv\_top** = (ImageView) view.findViewById(R.id.***iv\_top***);  
**tv\_search\_home** = (TextView) view.findViewById(R.id.***tv\_search\_home***);  
**tv\_message\_home** = (TextView) view.findViewById(R.id.***tv\_message\_home***);  
**iv\_top**.setOnClickListener(**this**);  
**tv\_search\_home**.setOnClickListener(**this**);  
**tv\_message\_home**.setOnClickListener(**this**);

@Override  
**public void** onClick(View view) **{  
  
 switch** (view.getId()) **{  
  
 case** R.id.***iv\_top***: Toast.makeText(**context**,"回顶部",Toast.***LENGTH\_SHORT***).show();  
 **recy\_home**.scrollToPosition(0);  
 **break**;  
  
 **case** R.id.***tv\_search\_home***: Toast.makeText(**context**,"搜索",Toast.***LENGTH\_SHORT***).show();  
 **break**;  
  
 **case** R.id.***tv\_message\_home***: Toast.makeText(**context**,"消息",Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 **}  
}**

// 不忘实现监听接口

**public class** HomeFragment **extends** BaseFragment **implements** View.OnClickListener {

}

### 使用OKHttpUtils联网请求数据

#### 项目地址

https://github.com/hongyangAndroid/okhttputils

#### 添加依赖

compile 'com.zhy:okhttputils:2.6.2'

#### 配置OkhttpClient

##### 在Application中调用initClient（）方法进行设置，参照下图



##### 在工程中新建Application类来完成上述要求的初始化

*// 作用：整个软件***public class** MyApplication **extends** Application **{** @Override  
 **public void** onCreate() **{  
 super**.onCreate();  
  
 initOKHttpClient();  
 **}  
  
 private void** initOKHttpClient() **{** OkHttpClient okHttpClient = **new** OkHttpClient.Builder()  
 *//.addInterceptor(new LoggerInterceptor("TAG"))* .connectTimeout(10000L, TimeUnit.***MILLISECONDS***)  
 .readTimeout(10000L, TimeUnit.***MILLISECONDS***)  
 *//其他配置* .build();  
  
 OkHttpUtils.initClient(okHttpClient);  
 **}  
}**

##### 在配置文件<application>中的首个位置声明

android:name=".app.MyApplication"

#### 使用OKHttpUtils，在HomeFragment中进行联网请求测试

##### 请求参考



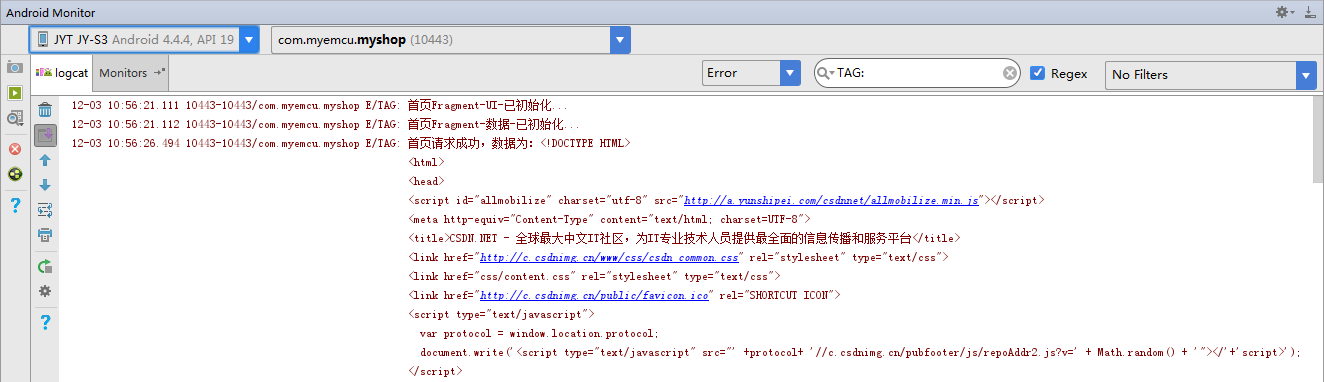
##### 具体实现

@Override  
**public void** initData() **{  
 super**.initData();  
 Log.e("TAG","首页Fragment-数据-已初始化...");  
  
 *// OKHttpUtils的Get请求* String url = "http://www.csdn.net/"; // 对该地址的数据进行联网请求  
 OkHttpUtils  
 .get()  
 .url(url)  
 */\*.addParams("username", "hyman")  
 .addParams("password", "123")\*/* .build()  
 .execute(**new** StringCallback()  
 **{** @Override *// 联网失败回调它* **public void** onError(Call call, Exception e, **int** id) **{** Log.e("TAG","首页请求失败，原因："+e.getMessage());  
 **}** @Override *// 联网成功回调它(response：请求成功的数据；id：区分 http(100) / https(101))* **public void** onResponse(String response, **int** id) **{** Log.e("TAG","首页请求成功，数据为："+response);  
 **}  
 }**);  
**}**

##### 添加权限

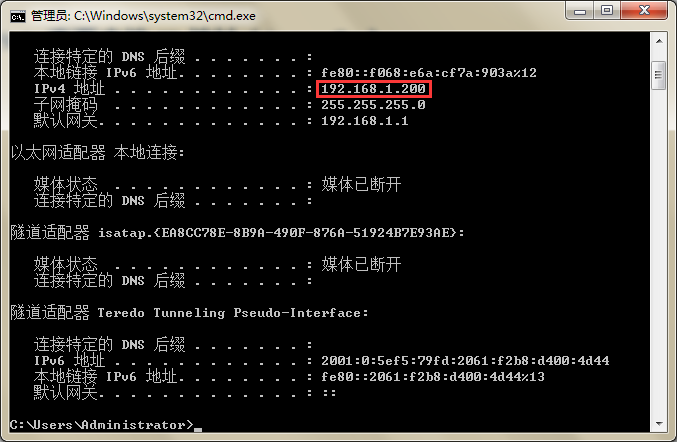
<**uses-permission** android:name="android.permission.INTERNET"/>

##### 测试结果（开WiFi）

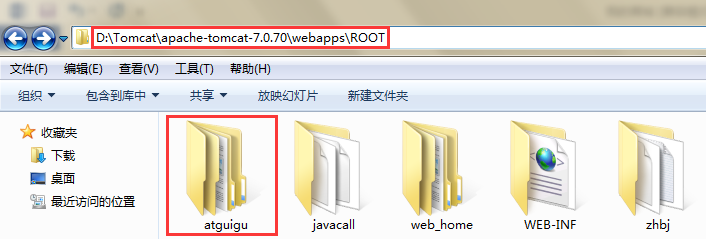


### 请求自己的数据

#### 查看本地IP地址（ipconfig）

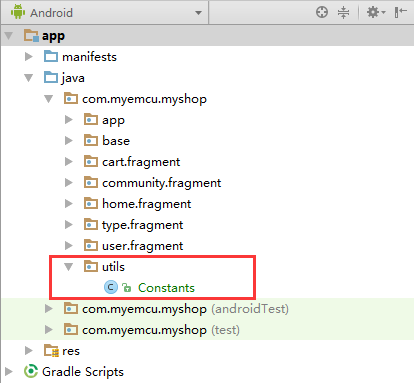


#### 将服务器资源文件复制到Tomcat的webapps\ROOT路径



#### 在工程中新建工具类-常量类

##### 工程目录树



##### 代码内容

*// 配置各页面联网地址***public class** Constants **{  
 public final static** String ***HOME\_URL*** = "http://192.168.1.200:8080/atguigu/json/HOME\_URL.json";  
**}**

#### 源码修改

String url = Constants.***HOME\_URL***;

#### 测试结果，略有延时（开WiFi、Tomcat）

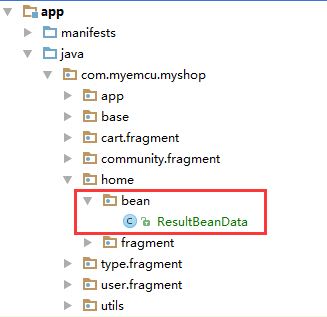


### FastJson解析数据

#### 添加依赖

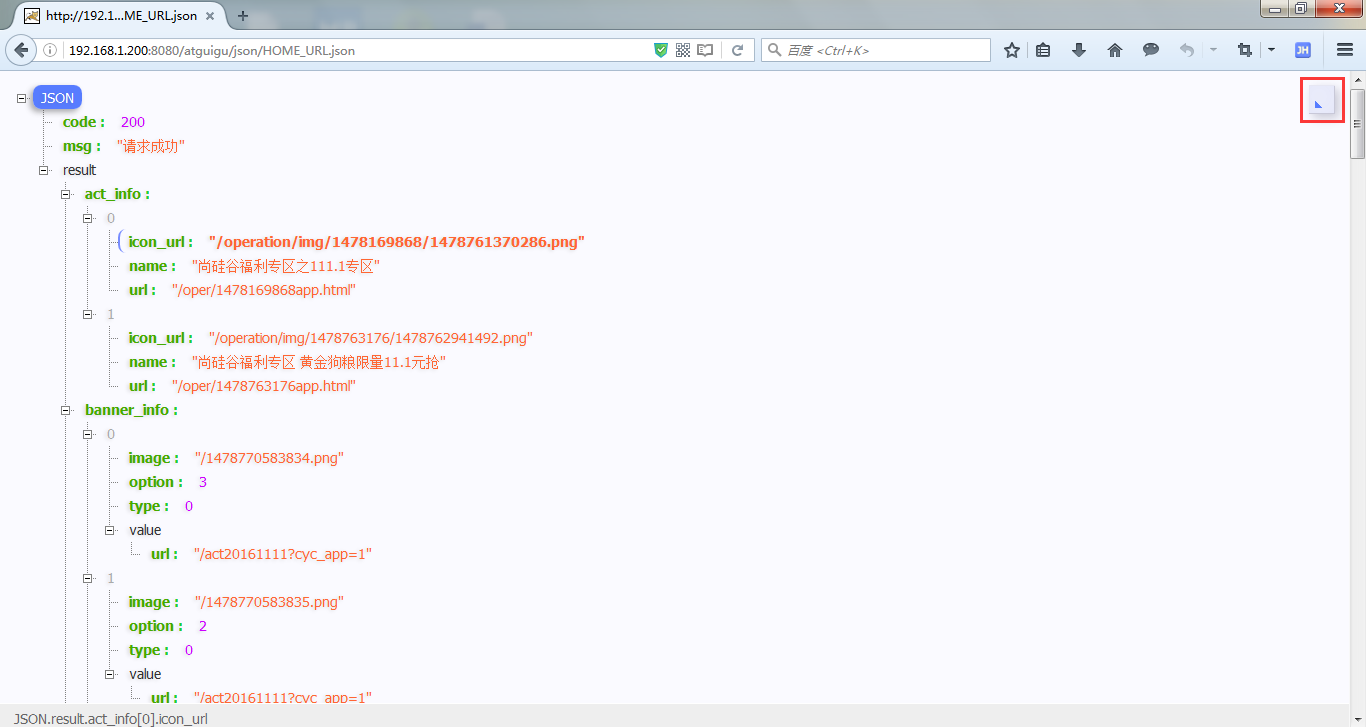
compile 'com.alibaba:fastjson:1.1.55.android'

#### 建立Bean对象



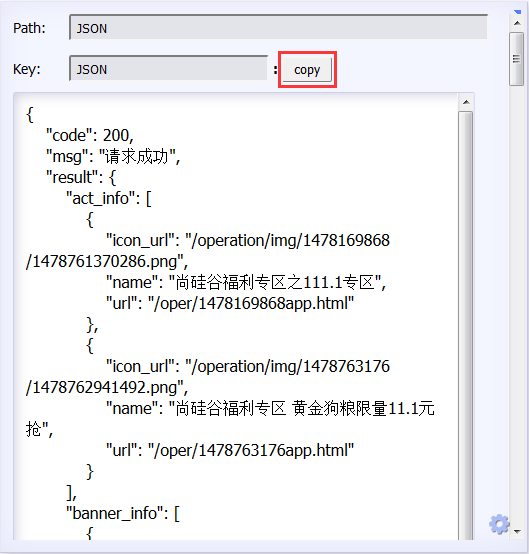
#### 在ResultBeanData中，用Gson（FastJson）生成bean代码（Alt+S）

##### IE(火狐)打开：<http://192.168.1.200:8080/atguigu/json/HOME_URL.json>

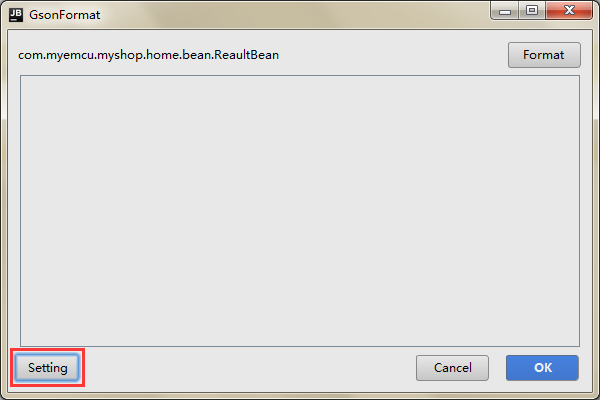


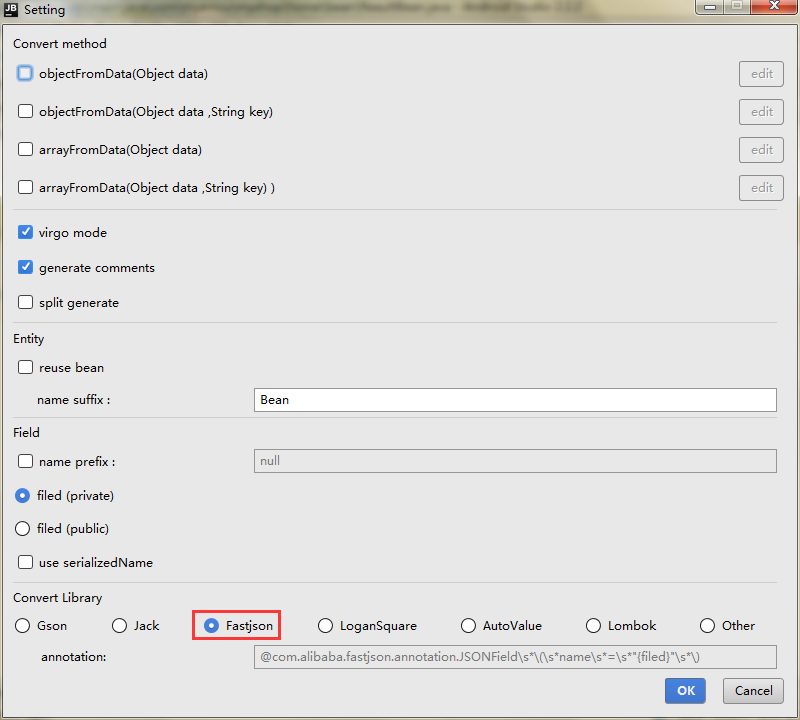
点这里

##### 展开Json(右上角)

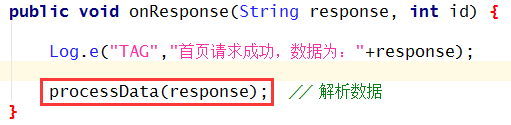


##### 将里面的内容复制到GsonFormat并使用FastJson生成Bean





#### 在HomeFragment中的联网成功处，解析数据



#### 解析代码（解析一条作测试）

**private void** processData(String json) **{** ResultBeanData resultBeanData = JSON.parseObject(json,ResultBeanData.**class**); *// 解析到后边的类中* **resultBean** = resultBeanData.getResult();  
 Log.e("TAG","解析成功=="+**resultBean**.getHot\_info().get(0).getName());  
**}**

#### 解析结果



### 首页UI

#### Json数据总览



#### UI出场顺序

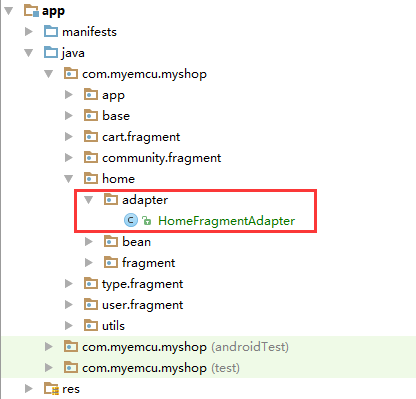
* 横幅（Banner） —— 0
* 频道（Channel） —— 1
* 活动（Act） —— 2
* 秒杀（Seckill） —— 3
* 推荐（Recommend） —— 4
* 热卖（Hot） —— 5

### 首页适配器(重点)

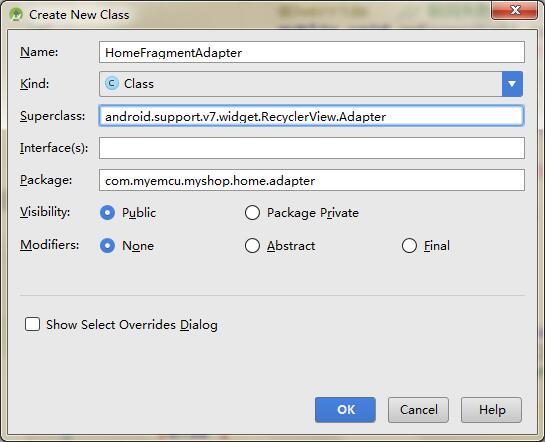
#### 总适配器(HomeFragmentAdapter)

一个基于HomeFragment的总适配器HomeFragmentAdapter，它继承自RecyclerView.Adapter。

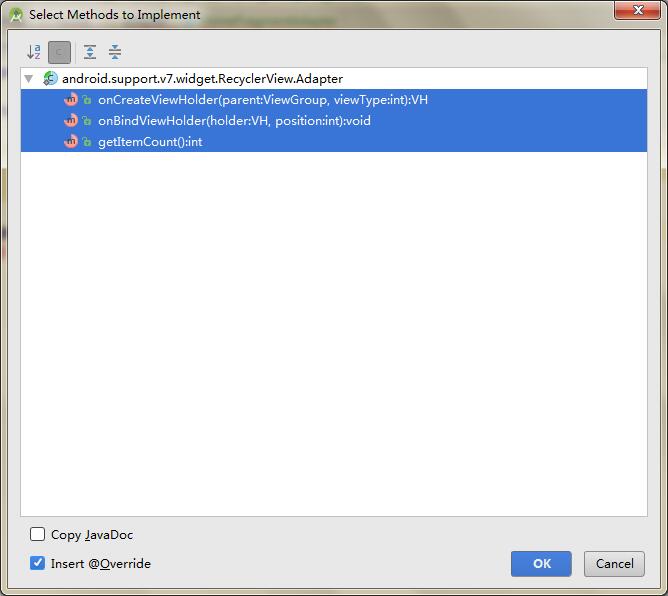
##### 建立适配器类



##### 继承对象



##### 实现方法



##### 原理分析

6种条目，对应6种RecyclerView，故需要一个继承于它的适配器：HomeFragmentAdapter，即：谁要显示RecyclerView，谁就需要适配器。

##### 实现步骤

1. 必须要实现的3个方法

@Override *// 创建ViewHolder(类似getView())***public** RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) **{  
 return null**;  
**}**@Override  
**public void** onBindViewHolder(RecyclerView.ViewHolder holder, **int** position) **{  
  
}**@Override *// 根据json可知，有6条item***public int** getItemCount() **{  
 return** 1;  
 *//return 6; // 一条条做，最后才是这个，为6就是返回0~5***}**

1. 变量定义

private final static int *BANNER* = 0; *// 横幅*private final static int *CHANNEL* = 1; *// 频道*private final static int *ACT* = 2; *// 活动*private final static int *SECKILL* = 3; *// 秒杀*private final static int *RECOMMEND* = 4; *// 推荐*private final static int *HOT* = 5; *// 热卖*private int currentType = *BANNER*; *// 定义当前条目的View类型(共6种)*

1. 补充实现的一个方法

@Override *// 获取条目的View类型(因为有6条)***public int** getItemViewType(**int** position) **{  
  
 switch** (position) **{  
 case *BANNER***: **currentType**=***BANNER***;  
 **break**;  
  
 **case *CHANNEL***: **currentType**=***CHANNEL***;  
 **break**;  
  
 **case *ACT***: **currentType**=***ACT***;  
 **break**;  
  
 **case *SECKILL***: **currentType**=***SECKILL***;  
 **break**;  
  
 **case *RECOMMEND***: **currentType**=***RECOMMEND***;  
 **break**;  
  
 **case *HOT***: **currentType**=***HOT***;  
 **break**;  
 **}  
  
 return currentType**;  
**}**

1. 到HomeFragment中的数据解析处设置该适配器

**private void** processData(String json) **{** ResultBeanData resultBeanData = JSON.parseObject(json,ResultBeanData.**class**); *// 解析到后边类中* **resultBean** = resultBeanData.getResult();  
 *//Log.e("TAG","解析成功=="+resultBean.getHot\_info().get(0).getName()); // 解析测试* **if** (**resultBean** != **null**) **{** *// 有数据，设置适配器* **adapter** = **new** HomeFragmentAdapter(**context**,**resultBean**); *// 传入上下文和数据* **recy\_home**.setAdapter(**adapter**);  
 **}else {  
  
 }  
}**

1. 适配器中的构造器

*// 构造器(上下文、数据、item)***public** HomeFragmentAdapter(Context context, ResultBeanData.ResultBean resultBean) **{  
 this**.**context**=context;  
 **this**.**resultBean**=resultBean;  
 **layoutInflater**=LayoutInflater.from(context); *// 用这个初始化布局***}**

1. 补充定义变量

*// 构造器接收***private final** Context **context**;  
**private final** ResultBeanData.ResultBean **resultBean**;  
*// 布局初始化***private final** LayoutInflater **layoutInflater**; *// 直接用View.inflate的性能消耗较大*

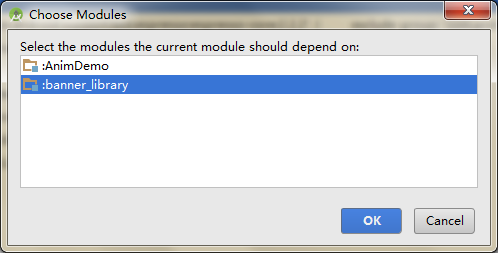
#### 横幅实现类(Banner)

##### 导入开源项目库banner

导入后名称改为banner\_library，路径：【…3 源码、课件\资料\1.第三方库\banner】



##### 添加Module依赖



##### 创建BannerViewHolder（即：创建ViewHolder）

@Override *// 创建ViewHolder(类似getView())，viewType来自getItemViewType(position)***public** RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) **{  
  
 if** (viewType == ***BANNER***) **{  
 return new** BannerViewHolder(**context**,**layoutInflater**.inflate(R.layout.***banner\_viewpager***,**null**));  
 **}  
  
 return null**;  
**}**

##### banner\_viewpager.xml

说明：命名空间app来自其库中的属性文件attr.xml

*<?*xml version="1.0" encoding="utf-8"*?>*<**LinearLayout** xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"   
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
 <**com.youth.banner.Banner** android:id="@+id/banner"  
 android:layout\_width="match\_parent"  
 android:layout\_height="130dp"  
 app:indicator\_height="5dp"  
 app:indicator\_width="5dp"  
 app:is\_auto\_play="true" />  
</**LinearLayout**>

##### 实现BannerViewHolder内部类（暂无数据，数据在后边的setData()中）

**private class** BannerViewHolder **extends** RecyclerView.ViewHolder **{  
 private final** Context **context**;  
 **private final** Banner **banner**;  
  
 // 构造器(传入：上下文、布局文件)

// (接收：上下文、布局中的控件)

**public** BannerViewHolder(Context context, View itemview) **{** *// itemview：传入的布局名* **super**(itemview);  
 **this**.**context**=context;  
 **this**.**banner**= (Banner) itemview.findViewById(R.id.***banner***); *// 接收布局中的控件* **}  
}**

##### 绑定BannerViewHolder（即：绑定ViewHolder）

@Override  
**public void** onBindViewHolder(RecyclerView.ViewHolder holder, **int** position) **{  
 if** (getItemViewType(position)==***BANNER***) **{** BannerViewHolder bannerViewHolder = (BannerViewHolder) holder;  
 bannerViewHolder.setData(**resultBean**.getBanner\_info()); // 传入Banner数据  
 **}  
}**

##### 显示数据showData()

**public void** setData(List<ResultBeanData.ResultBean.BannerInfoBean> banner\_info) **{** *// 建立一个String集合，用来装图片地址* List<String> imagesUrl = **new** ArrayList<>();  
  
 **for** (**int** i =0; i<banner\_info.size(); i++) **{** String url = banner\_info.get(i).getImage(); *// 得到每一条图片地址* imagesUrl.add(url);  
 **}  
  
 banner**.setBannerAnimation(Transformer.*Accordion*); *// 手风琴动画特效* **banner**.setBannerStyle(BannerConfig.***CIRCLE\_INDICATOR***); *// 横幅加点指示* **banner**.setImages(imagesUrl, **new** OnLoadImageListener() **{** @Override  
 **public void** OnLoadImage(ImageView view, Object url) **{** *// Glide联网请求图片(使用jar包)  
 // 说明：此处将banner库中的glide版本由3.6.1->3.7.0，才能烧写通过，不然就报多重定义错* Glide.with(**context**).load(Constants.***BASE\_URL\_IMAGE*** + url).into(view);  
 **}  
 }**);  
 **banner**.setOnBannerClickListener(**new** OnBannerClickListener() **{** *// Banner点击事件* @Override  
 **public void** OnBannerClick(**int** position) **{** Toast.makeText(**context**,"你点了第"+position+"条横幅",Toast.***LENGTH\_SHORT***).show();  
 **}  
 }**);  
**}**

##### 设置HomeFeagment中的RecyclerView布局方式

**recy\_home**.setLayoutManager(**new** GridLayoutManager(**context**,1)); *// 1列*

##### 实现效果



##### 编程思想

HomeFragmentAdapter适配器的编写流程

1. 在需要用到该适配器的文件中（HomeFragment）将其new出并设置

**adapter** = **new** HomeFragmentAdapter(**context**,**resultBean**); *// 传入上下文和数据***recy\_home**.setAdapter(**adapter**);  
**recy\_home**.setLayoutManager(**new** GridLayoutManager(**context**,1)); *// RecyclerView还需设置布局样式*

1. 确定该适配器应当继承哪个

**public class** HomeFragmentAdapter **extends** RecyclerView.Adapter **{**

**}**

1. 根据传入参数编写构造器

*// 构造器***public** HomeFragmentAdapter(Context context, ResultBeanData.ResultBean resultBean) **{  
 this**.**context**=context;  
 **this**.**resultBean**=resultBean;  
  
 **layoutInflater**=LayoutInflater.from(context); *// 根据传入的上下文，创建该适配器的布局加载对象***}**

1. 必须实现的3个方法和补充实现的一个方法
2. 创建ViewHolder——onCreateViewHolder(ViewGroup parent, int viewType)；
3. 绑定ViewHolder——onBindViewHolder(RecyclerView.ViewHolder holder, int position)；
4. item条目数量；
5. getItemViewType(int position)——根据位置，设置当前条目类型（从0开始），它的返回结果会回传到onCreateViewHolder的int viewType中；
6. 真正实现Banner显示的类：**BannerViewHolder()。**

各自代码如下：

**onCreateViewHolder()**

@Override *// 创建ViewHolder(类似getView())，viewType来自getItemViewType(position)***public** RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) **{  
  
 if** (viewType == ***BANNER***) **{** *// 传入上下文、布局(itemview)* **return new BannerViewHolder**(**context**, **layoutInflater**.inflate(R.layout.***banner\_viewpager***,**null**));  
 **}  
  
 return null**;  
**}**

**onBindViewHolder()**

@Override *// 绑定ViewHolder***public void** onBindViewHolder(RecyclerView.ViewHolder holder, **int** position) **{**

**if** (getItemViewType(position)==***BANNER***) **{** BannerViewHolder bannerViewHolder = (BannerViewHolder) holder;  
 bannerViewHolder.showData(**resultBean**.getBanner\_info()); *// 传入Banner数据* **}  
}**

**getItemCount()**

@Override *// 根据json可知，有6条item***public int** getItemCount() **{  
 return** 1;  
 *//return 6; // 一条条做，最后才是这个，为6就是返回0~5***}**

**getItemViewType()**

@Override *// 获取条目的View类型(因为有6条)***public int** getItemViewType(**int** position) **{  
  
 switch** (position) **{  
 case *BANNER***: **currentType**=***BANNER***;  
 **break**;  
  
 **case *CHANNEL***: **currentType**=***CHANNEL***;  
 **break**;  
  
 **case *ACT***: **currentType**=***ACT***;  
 **break**;  
  
 **case *SECKILL***: **currentType**=***SECKILL***;  
 **break**;  
  
 **case *RECOMMEND***: **currentType**=***RECOMMEND***;  
 **break**;  
  
 **case *HOT***: **currentType**=***HOT***;  
 **break**;  
 **}  
  
 return currentType**;  
**}**

**BannerViewHolder()**

**private class** BannerViewHolder **extends** RecyclerView.ViewHolder **{**

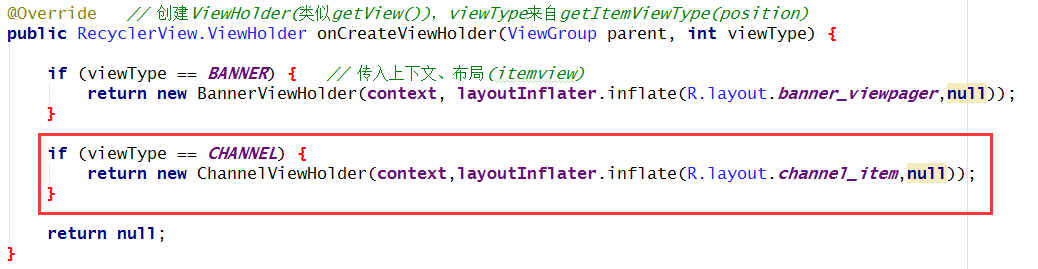
**private final** Context **context**;  
 **private final** Banner **banner**;  
  
 *// 构造器(传入：上下文、布局文件)  
 // (接收：上下文、布局中的控件)* **public** BannerViewHolder(Context context, View itemview) **{***// itemview：传入的布局名*

**super**(itemview);  
 **this**.**context**=context;  
 **this**.**banner**= (Banner) itemview.findViewById(R.id.***banner***);*// 接收布局中的控件* **}  
  
 public void** showData(List<ResultBeanData.ResultBean.BannerInfoBean> banner\_info) **{** *// 建立一个String集合，用来装图片地址* List<String> imagesUrl = **new** ArrayList<>();  
  
 **for** (**int** i =0; i<banner\_info.size(); i++) **{** String url = banner\_info.get(i).getImage(); *// 得到每一条图片地址* imagesUrl.add(url);  
 **}  
  
 banner**.setBannerAnimation(Transformer.*Accordion*); *// 手风琴动画特效* **banner**.setBannerStyle(BannerConfig.***CIRCLE\_INDICATOR***); *// 横幅加点指示* **banner**.setImages(imagesUrl, **new** OnLoadImageListener() **{** @Override  
 **public void** OnLoadImage(ImageView view, Object url) **{** *// Glide联网请求图片(使用jar包)  
 // 说明：此处将banner库中的glide版本由3.6.1->3.7.0，才能烧写通过，不然就报多重定义错* Glide.with(**context**).load(Constants.***BASE\_URL\_IMAGE*** + url).into(view);  
 **}  
 }**);

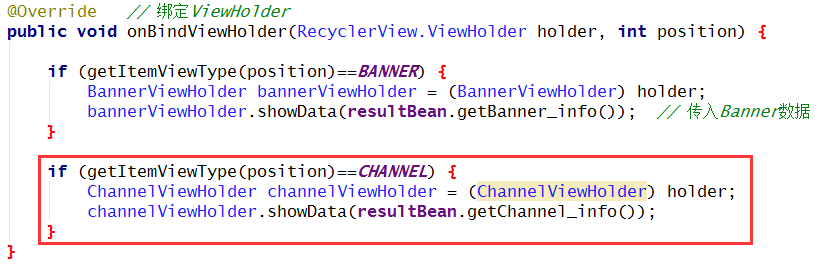
**banner**.setOnBannerClickListener(**new** OnBannerClickListener() **{** *// Banner点击事件* @Override  
 **public void** OnBannerClick(**int** position) **{** Toast.makeText(**context**,"你点了第"+position+"条横幅",Toast.***LENGTH\_SHORT***).show();  
 **}  
 }**);  
 **}  
}**

#### 频道适配器(Channel)

##### 在onCreateViewHolder()中追加



##### 在onBindViewHolder()中追加



##### ChannelViewHolder实现类

**private class** ChannelViewHolder **extends** RecyclerView.ViewHolder **{  
 private final** Context **context**;  
 **private** GridView **gv\_channel**;  
  
 **private** ChannelAdapter **adapter**;  
  
 **public** ChannelViewHolder(Context context, View itemView) **{  
 super**(itemView);  
  
 **this**.**context**=context;  
 **gv\_channel**= (GridView) itemView.findViewById(R.id.***gv\_channel***);  
 **}  
  
 public void** showData(**final** List<ResultBeanData.ResultBean.ChannelInfoBean> channel\_info) **{** *// 数据过来了，需要设置GridView适配器* **adapter** = **new** ChannelAdapter(**context**,channel\_info); *// 传入上下文和数据*

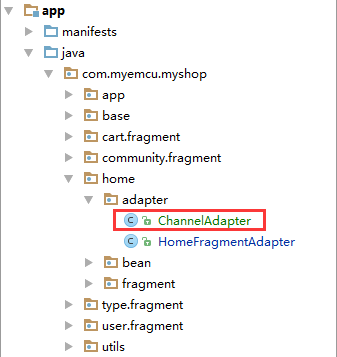
**gv\_channel**.setAdapter(**adapter**);  
 **gv\_channel**.setOnItemClickListener(**new** AdapterView.OnItemClickListener() **{** @Override  
 **public void** onItemClick(AdapterView<?> view, View view1, **int** position, **long** l) **{** String content = channel\_info.get(position).getChannel\_name();  
 Toast.makeText(**context**,"你点击的是："+content,Toast.***LENGTH\_SHORT***).show();  
 **}  
 }**);  
 **}  
}**

##### ChannelAdapter类

**public class** ChannelAdapter **extends** BaseAdapter **{** *// GridView的适配器，用的是BaseAdapter* **private final** Context **context**;  
 **private final** List<ResultBeanData.ResultBean.ChannelInfoBean> **channel\_info**;  
  
 **public** ChannelAdapter(Context context, List<ResultBeanData.ResultBean.ChannelInfoBean>

channel\_info) **{  
 this**.**context**=context;  
 **this**.**channel\_info**=channel\_info;  
 **}** @Override  
 **public int** getCount() **{  
 return channel\_info**.size();  
 **}** @Override  
 **public** Object getItem(**int** i) **{  
 return null**;  
 **}** @Override  
 **public long** getItemId(**int** i) **{  
 return** 0;  
 **}  
  
 private static class** ViewHolder **{  
 private** ImageView **iv\_channel**;  
 **private** TextView **tv\_channel**;  
 **}** @Override  
 **public** View getView(**int** position, View convertView, ViewGroup parent) **{** ViewHolder viewHolder;  
  
 **if** (convertView==**null**) **{** convertView = View.inflate(**context**,R.layout.***item\_channel***,**null**);  
 viewHolder = **new** ViewHolder();  
 viewHolder.**iv\_channel** = (ImageView) convertView.findViewById(R.id.***iv\_channel***);  
 viewHolder.**tv\_channel** = (TextView) convertView.findViewById(R.id.***tv\_channel***);  
 convertView.setTag(viewHolder);  
 **}else {** viewHolder = (ViewHolder) convertView.getTag();  
 **}** *// 显示图片* Glide.with(**context**)  
 .load(Constants.***BASE\_URL\_IMAGE***+**channel\_info**.get(position).getImage())  
 .into(viewHolder.**iv\_channel**);  
 *// 显示文字* viewHolder.**tv\_channel**.setText(**channel\_info**.get(position).getChannel\_name());  
  
 **return** convertView; *// 千万不要忘了这* **}  
}**

##### 工程目录



##### 实现效果

