# 综合实验3 《小吃APP》新品页面设计与功能实现

## 一、实验简介

本次实验将在上次实验代码的基础上实现【新品页面】设计及其功能逻辑，页面上分为2部分：App宣传栏、今日新品小吃列表。其主要利用RecyclerView + StaggeredGridLayout技术实现上下滑动的流式布局列表效果；实现用流式列表形式展示新品小吃。

## 二、实验目标

* 掌握RecyclerView列表技术的使用步骤
* 掌握RecyclerView适配器类的定义
* 实现【新品页面】的排版布局

## 实验操作步骤

### 实现效果

### 实现步骤

#### 第1步：打开实验2开发的【XC】项目；

点击Android Studio工具左上角的【File】->【Open】->在弹出的界面里选择自己存放实验2项目【XC】的位置，选择【XC】，再点击【OK】即可。

#### 第2步：调整根布局代码

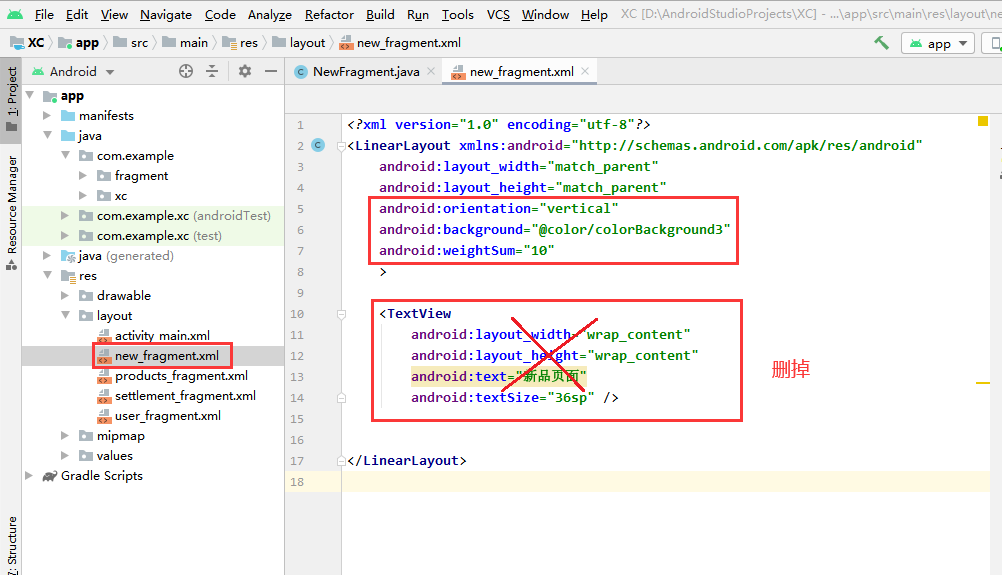
在<app\res\layout\>目录下的【new\_fragment.xml】布局文件里插入下面的代码，其作用是：把LinearLayout布局设置为上下结构形式，设置背景颜色，把整个布局或页面分成10等分。

android:orientation="vertical"

android:background="@color/colorBackground3"

android:weightSum="10"

上述代码插入或编辑的具体位置如下图所示



#### 第3步：在布局里添加ImageView控件

在<app\res\layout\>目录下的【new\_fragment.xml】布局文件里以拖拽形式或手动编写代码形式插入下面的代码，其效果为：用于展示店铺理念或特色，其大小占整个布局的3.1等分。

<ImageView

android:id="@+id/imageView"

android:layout\_width="match\_parent"

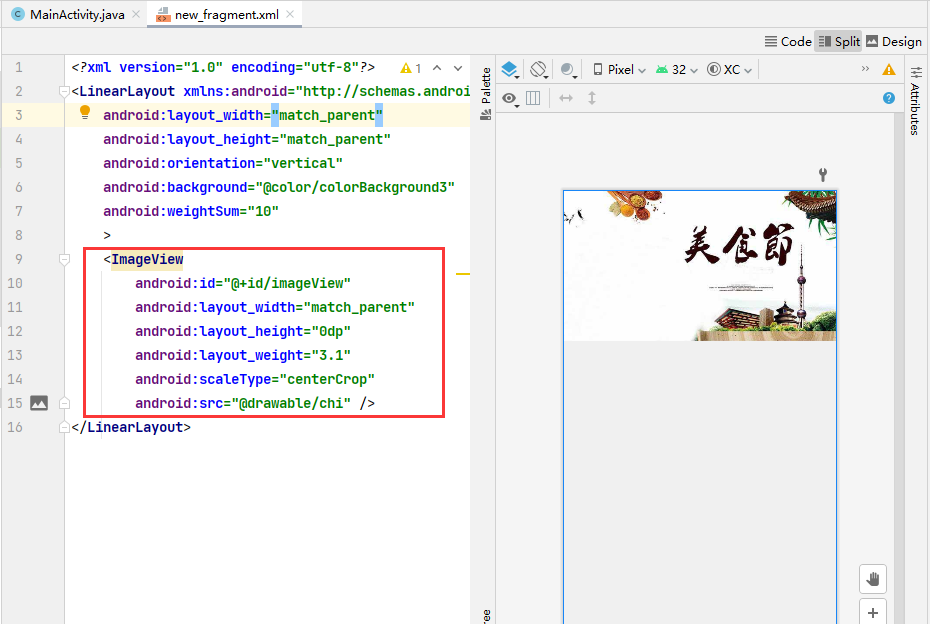
android:layout\_height="0dp"

android:layout\_weight="3.1"

android:scaleType="centerCrop"

android:src="@drawable/chi" />

上述代码插入的具体位置和效果如下图所示



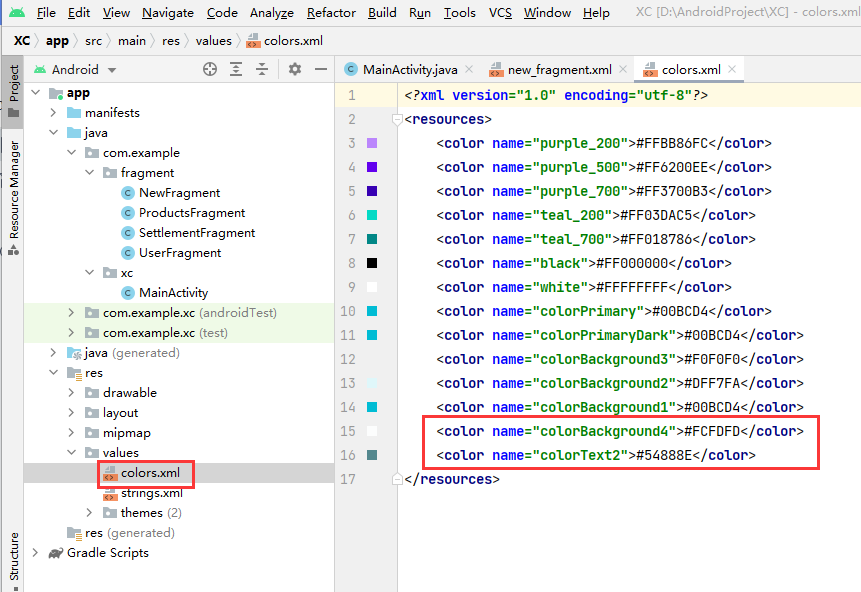
#### 第4步：在布局里添加CardView布局

先在<app\res\values\>目录下的【colors.xml】文件里插入下面的代码，其效果为：定义一种背景色，和一种文字颜色。

<color name="colorBackground4">#FCFDFD</color>

<color name="colorText2">#54888E</color>

上述代码插入的具体位置如下图所示



然后在<app\res\layout\>目录下的【new\_fragment.xml】布局文件里用拖拽形式或手动编写代码形式插入下面的代码，其作用为：以卡片效果呈现“今日新品”标题字样，其大小占整个布局的0.7等分。

<androidx.cardview.widget.CardView

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="0.7"

android:layout\_marginLeft="7dp"

android:layout\_marginRight="7dp"

android:layout\_marginTop="7dp"

android:layout\_marginBottom="7dp"

app:cardElevation="3dp"

>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:text="-- 今日新品 --"

android:gravity="center"

android:textSize="20sp"

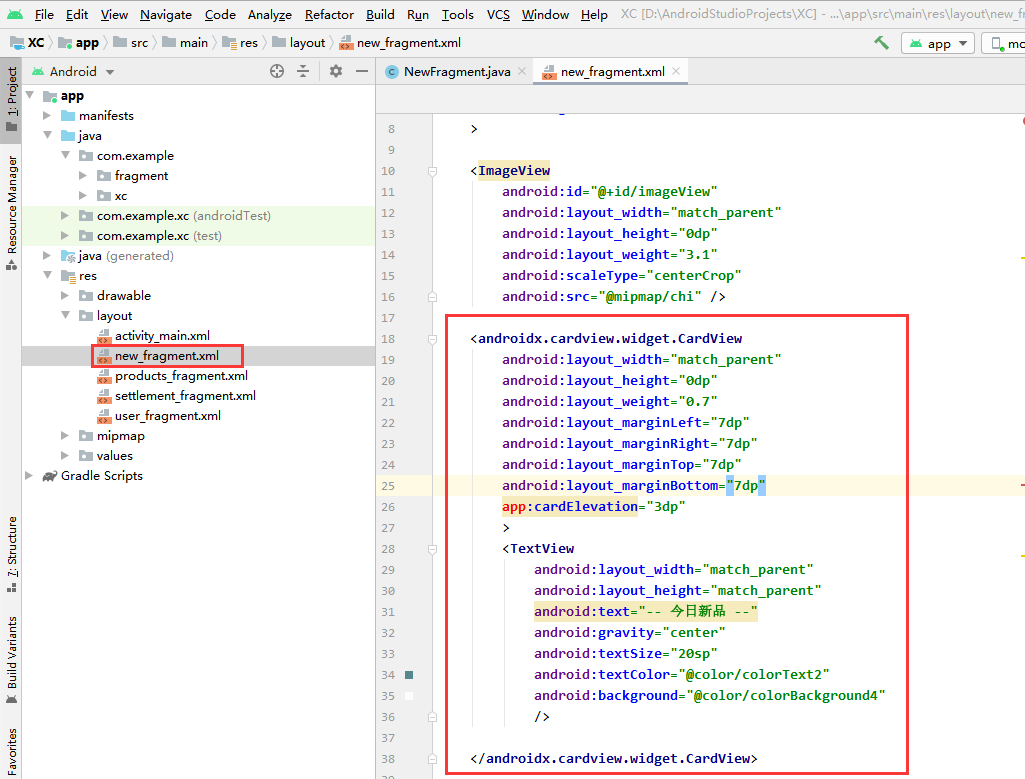
android:textColor="@color/colorText2"

android:background="@color/colorBackground4"

/>

</androidx.cardview.widget.CardView>

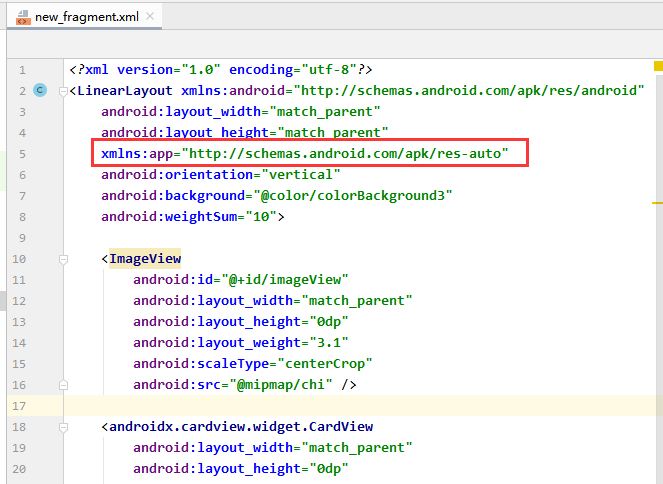
上述代码插入的具体位置如下图所示



最后，如果如上图中的“app”文字为大红色提示，则需要导入相应资源。导入的方式为：在根布局LinearLayout控件标签里插入下面的代码即可。

xmlns:app="http://schemas.android.com/apk/res-auto"

上述代码插入的具体位置如下图所示

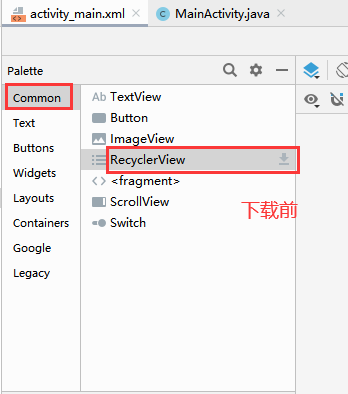
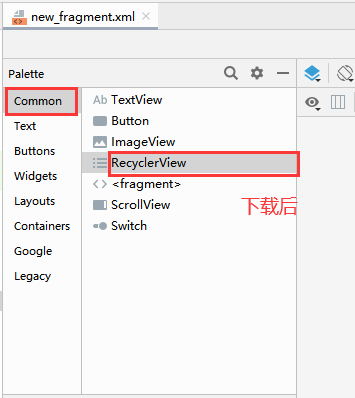


整个过程完成后，其效果如下所示。



#### 第5步：在布局里添加RecyclerView控件

先看控件栏里的RecyclerView控件有没有下载，没有下载就点击下载箭头下载。

再在上一步代码的后面，以拖拽形式或手动编写代码形式添加下面的代码，其作用为：添加RecyclerView列表到布局中，其大小占整个布局6.2等分。

<androidx.recyclerview.widget.RecyclerView

android:id="@+id/newRecycler"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

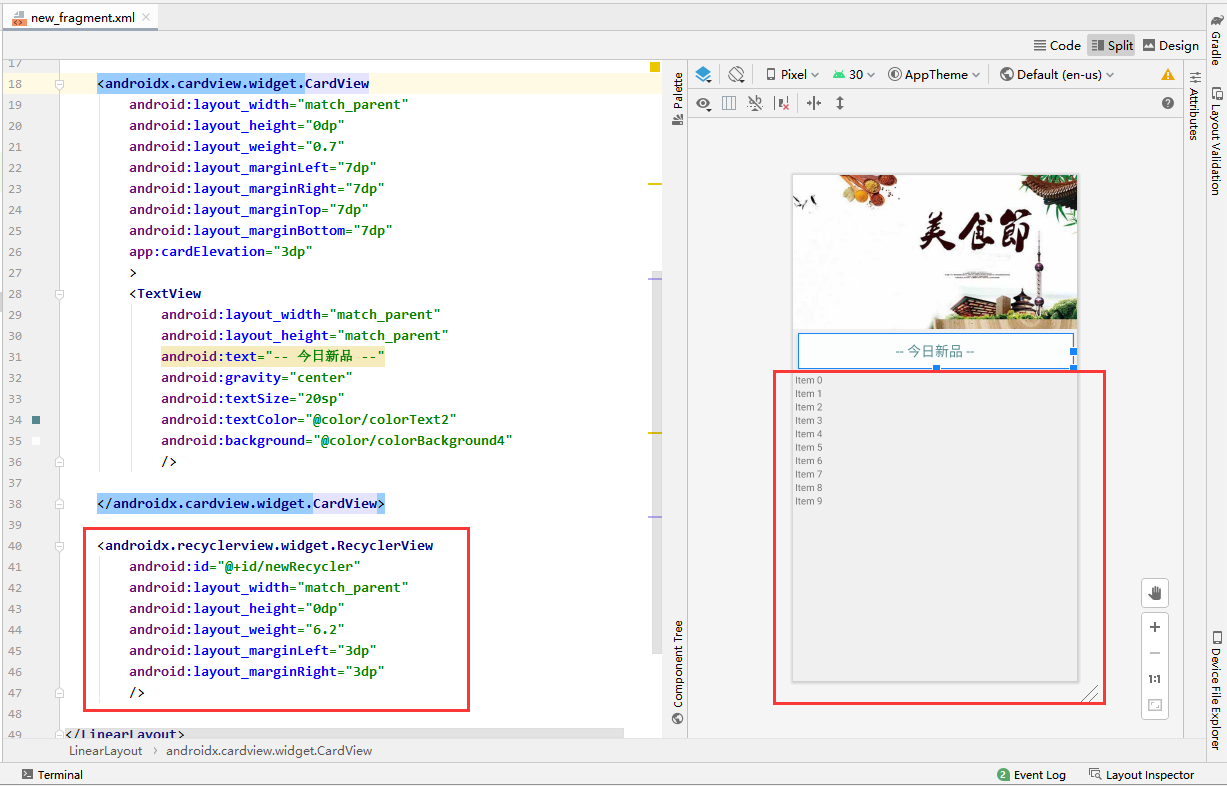
android:layout\_weight="6.2"

android:layout\_marginLeft="3dp"

android:layout\_marginRight="3dp"

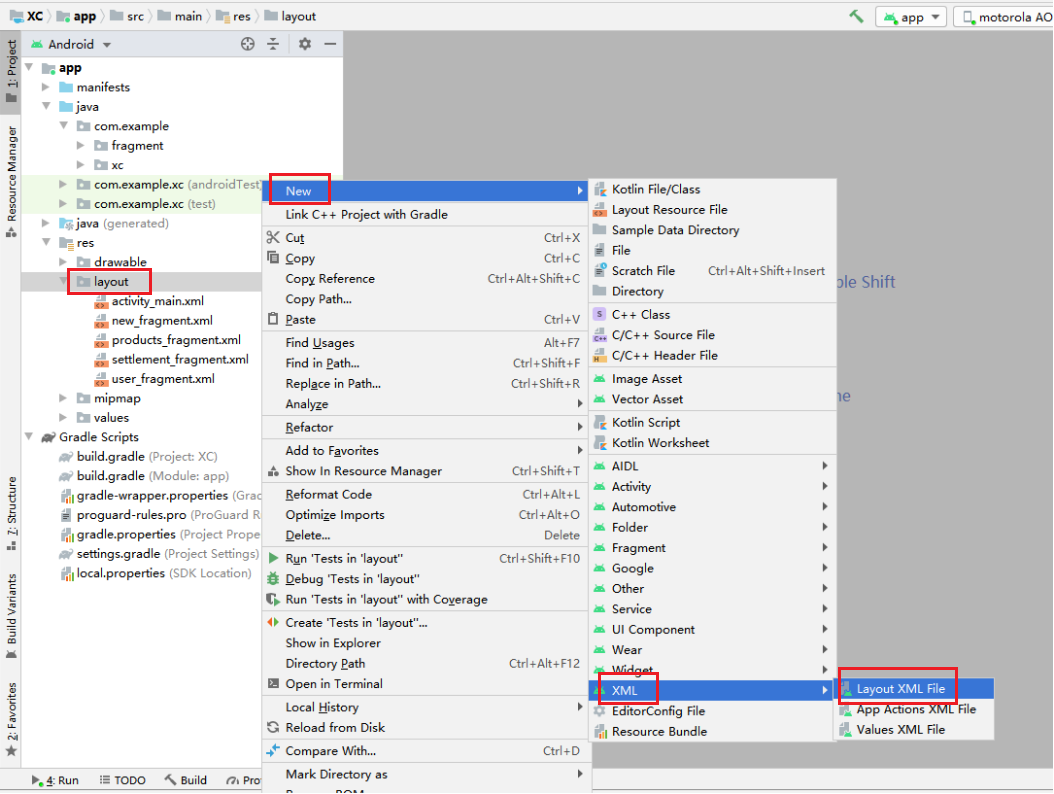
/>

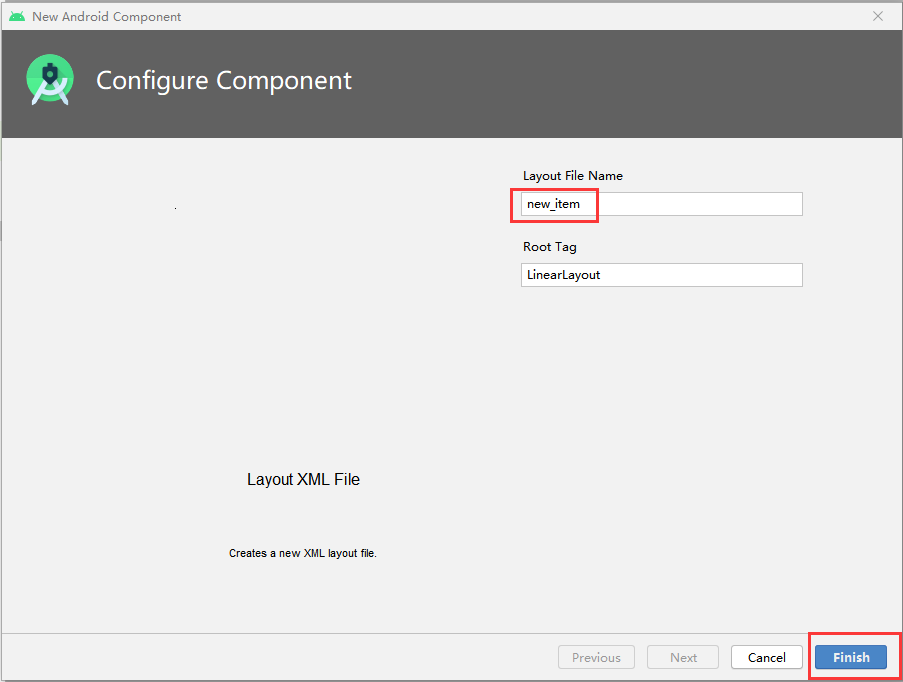
上述代码插入的具体位置和效果如下图所示



#### 第6步：新建列表选项布局文件

鼠标右键点击<app\res\layout\>目录，依次选择【New】->【XML】->【Layout XML File】->在弹出的窗口里填入文件名【new\_item】->再选择【Finish】即会自动创建【new\_item.xml】文件。





然后打开创建的【new\_item.xml】文件，插入或编辑为下面的两段代码，其作用为：把布局改为卡片效果，高度随列表选项内容大小。

androidx.cardview.widget.CardView

android:layout\_height="wrap\_content"

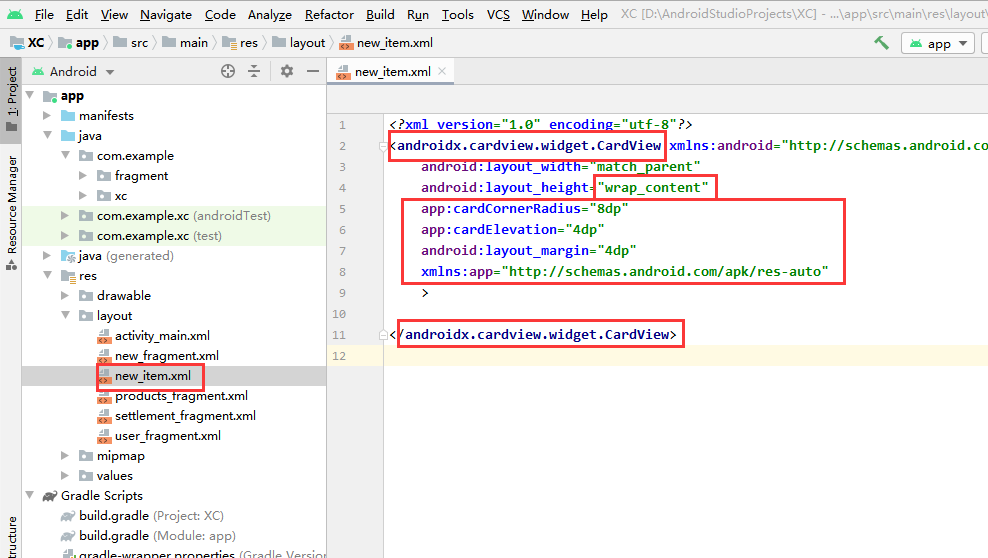
app:cardCornerRadius="8dp"

app:cardElevation="4dp"

android:layout\_margin="4dp"

xmlns:app="http://schemas.android.com/apk/res-auto"

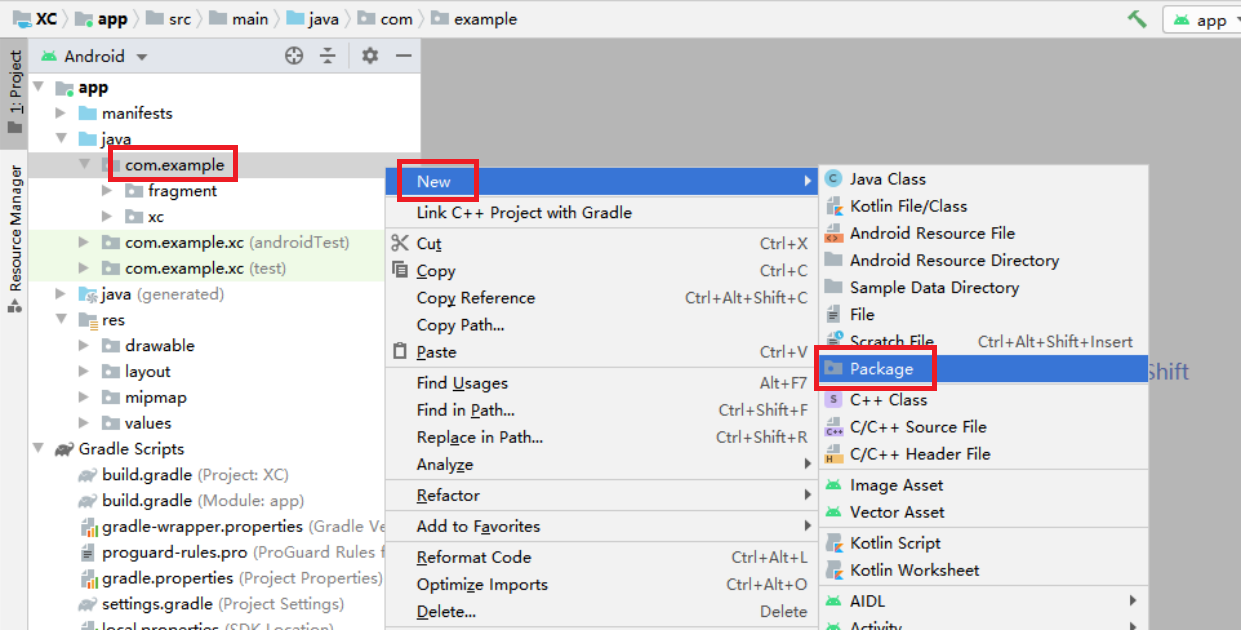
上述代码插入或编辑的具体位置如下图所示

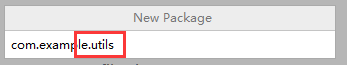


其中【app:cardCornerRadius="8dp"】用于设置控件边框的弧度，值越大弧度越大；【app:cardElevation="4dp"】用于设置阴影大小，值越大阴影越大。

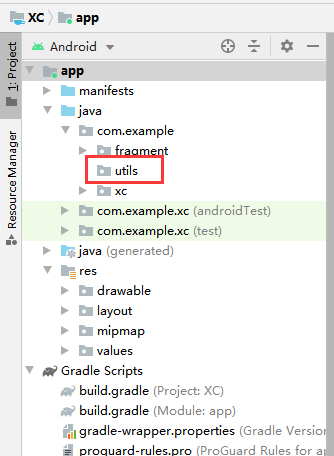
#### 第7步：新建包名utils

鼠标右键点击<app\java\com.example\>目录，依次选择【New】->【Package】->在弹出框里原有基础上添加包名【utils】回车即可。



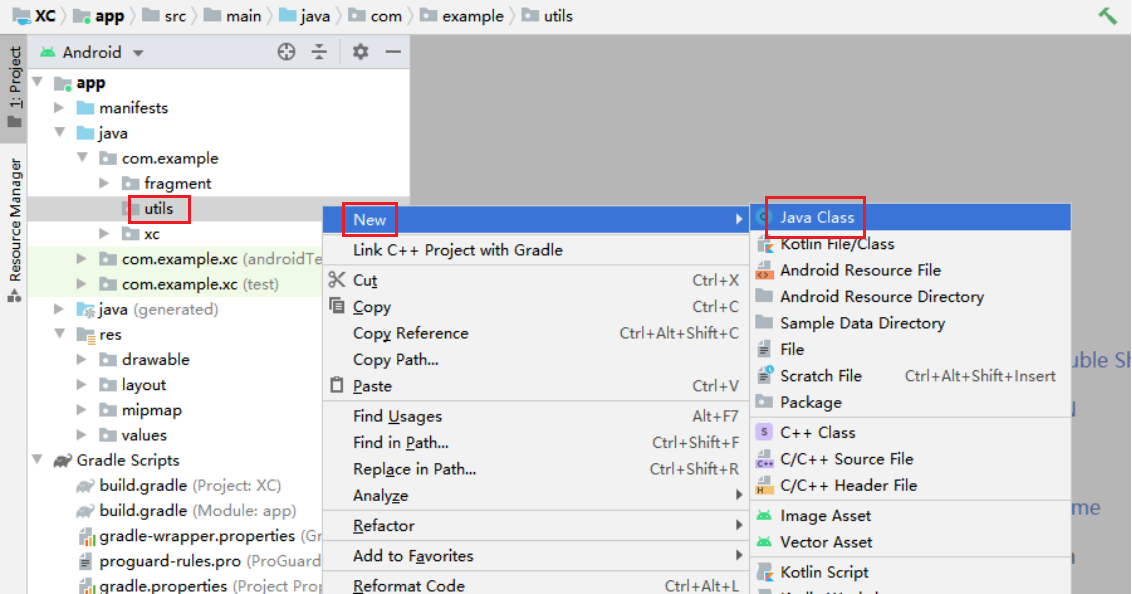


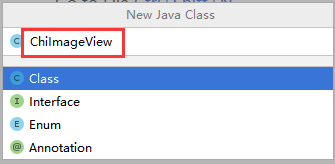
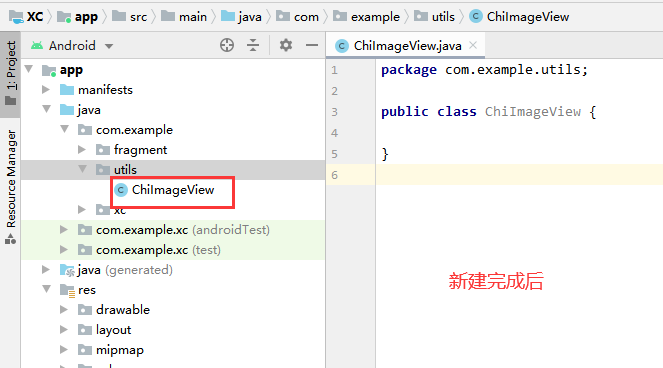
新建成功后的效果如下图所示



#### 第8步：自定义ChiImageView控件类

先在上一步新建的【utils】目录下新建一个名为【ChiImageView】的类，其步骤为：鼠标右键点击【utils】目录，依次选择【New】->【Java Class】->在弹出框里填入类名【ChiImageView】回车即可。



然后再把类编辑为下面的代码，其效果为：用ChiImageView呈现图片时，高度值会随宽度值的比例调整。

package com.example.utils;

import android.content.Context;

import android.graphics.drawable.Drawable;

import android.util.AttributeSet;

import androidx.annotation.NonNull;

import androidx.annotation.Nullable;

public class ChiImageView extends androidx.appcompat.widget.AppCompatImageView {

public ChiImageView(@NonNull Context context) {

super(context);

}

public ChiImageView(@NonNull Context context, @Nullable AttributeSet attrs) {

super(context, attrs);

}

public ChiImageView(@NonNull Context context, @Nullable AttributeSet attrs, int defStyleAttr) {

super(context, attrs, defStyleAttr);

}

@Override

protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec){

Drawable d = getDrawable();

if(d!=null){

int width = MeasureSpec.getSize(widthMeasureSpec);

//高度根据宽度计算

int height = (int) Math.ceil((float) width \* (float) d.getIntrinsicHeight() / (float) d.getIntrinsicWidth());

setMeasuredDimension(width, height);

}else{

super.onMeasure(widthMeasureSpec, heightMeasureSpec);

}

}

}

#### 第9步：设计列表选项布局

在第6步实现的【new\_item.xml】文件代码里插入下面的代码，其效果为：让选项布局里可以显示图片信息和文本信息，以上下结构排列，上面显示图片，下面显示文本。

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<com.example.utils.ChiImageView

android:id="@+id/newImage"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

app:srcCompat="@mipmap/ic\_launcher" />

<TextView

android:id="@+id/newText"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textSize="16sp"

android:gravity="center"

android:padding="5dp"

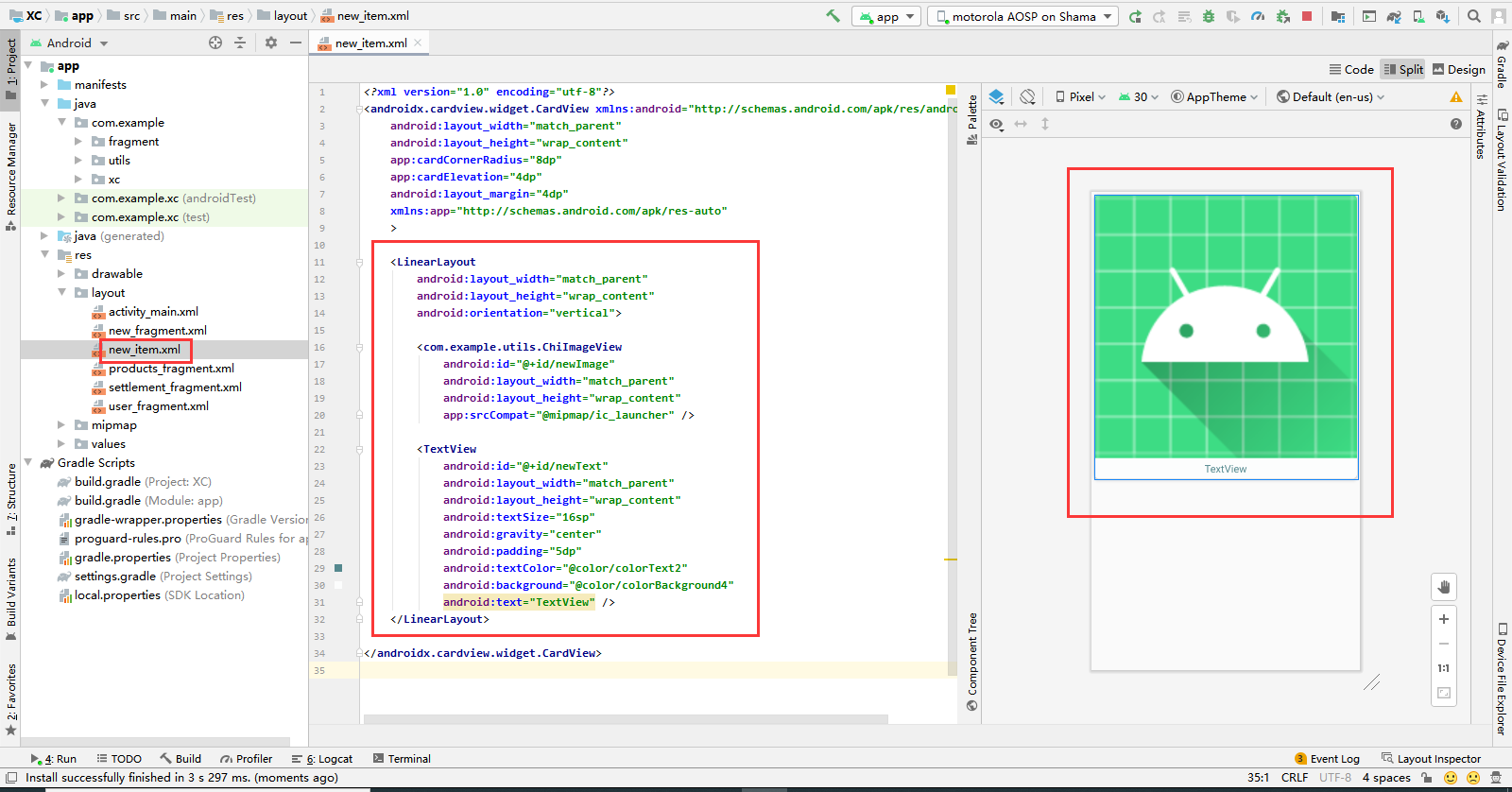
android:textColor="@color/colorText2"

android:background="@color/colorBackground4"

android:text="TextView" />

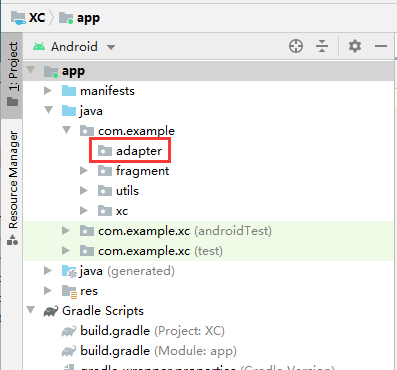
</LinearLayout>

上述代码插入的具体位置和效果如下图所示



#### 第10步：新建适配器类

先在<app\java\com.example\>目录下新建一个名字为【adapter】的包，具体步骤如第7步所示，新建完成后的效果如下图所示。



然后再在该目录下新建一个名为【NewAdapter】的类，新建的具体步骤如第8步所示，建好后，把类编辑为下面的代码。其作用为：适配器类是RecyclerView列表控件实现功能时必须的一个步骤，用于绑定选项数据。

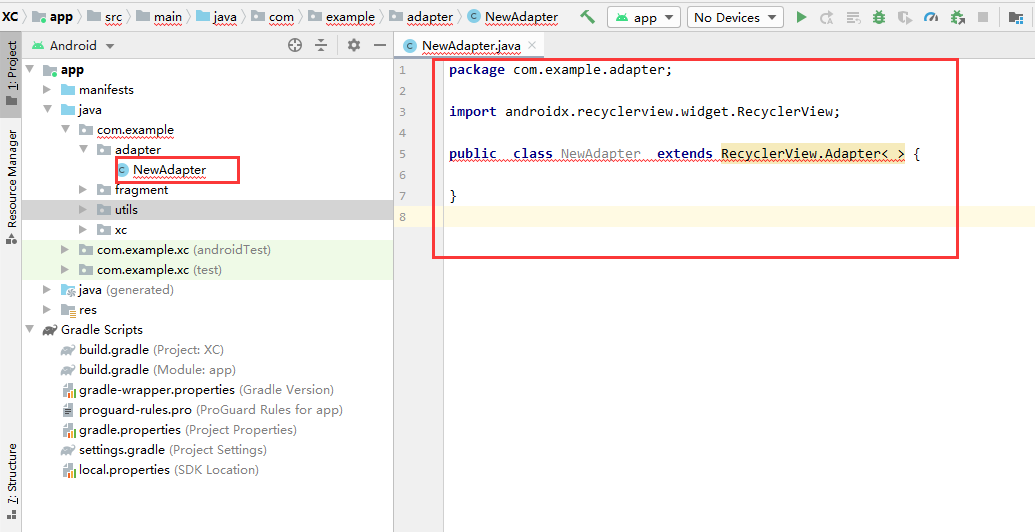
package com.example.adapter;

import androidx.recyclerview.widget.RecyclerView;

public class NewAdapter extends RecyclerView.Adapter< > {

}

上述步骤完成后的效果如下图所示



其中，报红色线的地方是由于代码还没有完全完成，到后面实现完后就没有了。

#### 第11步：定义适配器ViewHolder内部类

在上一步定义的适配类里插入下面两段代码，其作用是：定义内部类，用于缓存列表选项控件对象。

NewAdapter.ViewHolder

public static class ViewHolder extends RecyclerView.ViewHolder{

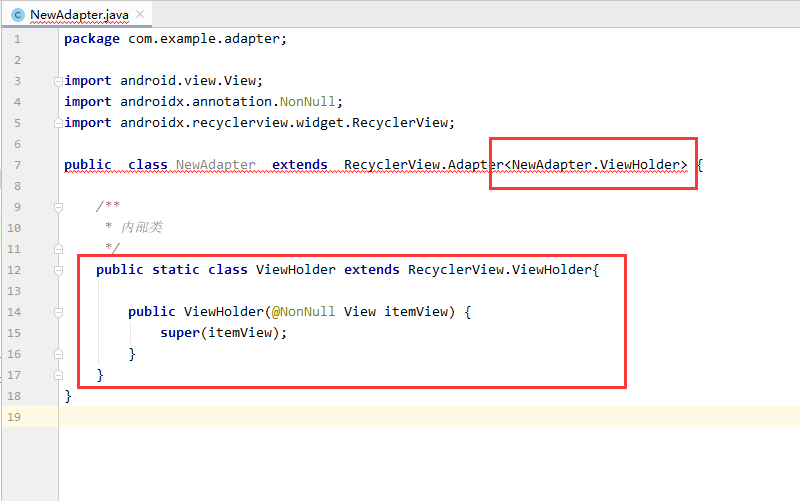
public ViewHolder(@NonNull View itemView) {

super(itemView);

}

}

上述代码插入的具体位置如下图所示



其中，报红色线的地方是由于代码还没有完全完成，到后面实现完后就没有了。上述代码在编写时，需要对View类和@NonNull注解导包，需要导入

import android.view.View;

import androidx.annotation.NonNull;

#### 第12步：重写适配器3个方法

在上一步代码的基础上，插入下面的代码，其作用为：适配器类必需要重写的3个方法，用于执行适配器的相关流程。

@NonNull

@Override

public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

return null;

}

@Override

public void onBindViewHolder(@NonNull ViewHolder holder, int position) {

}

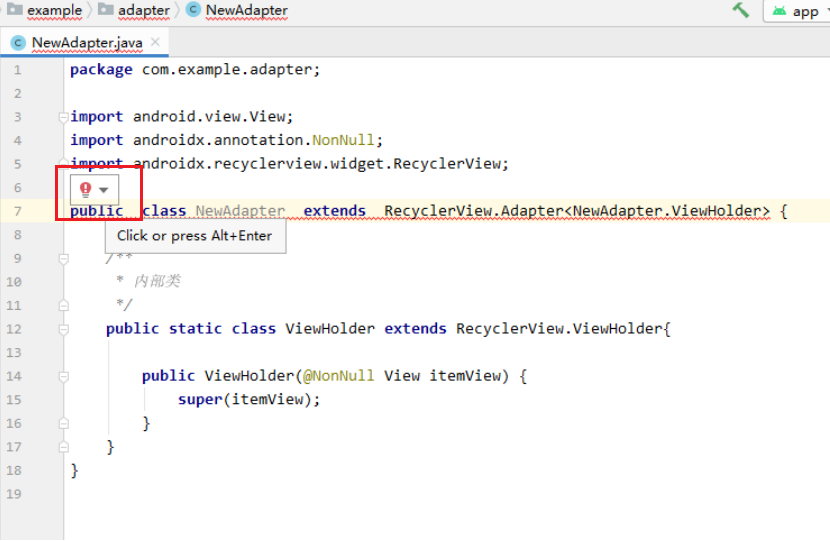
@Override

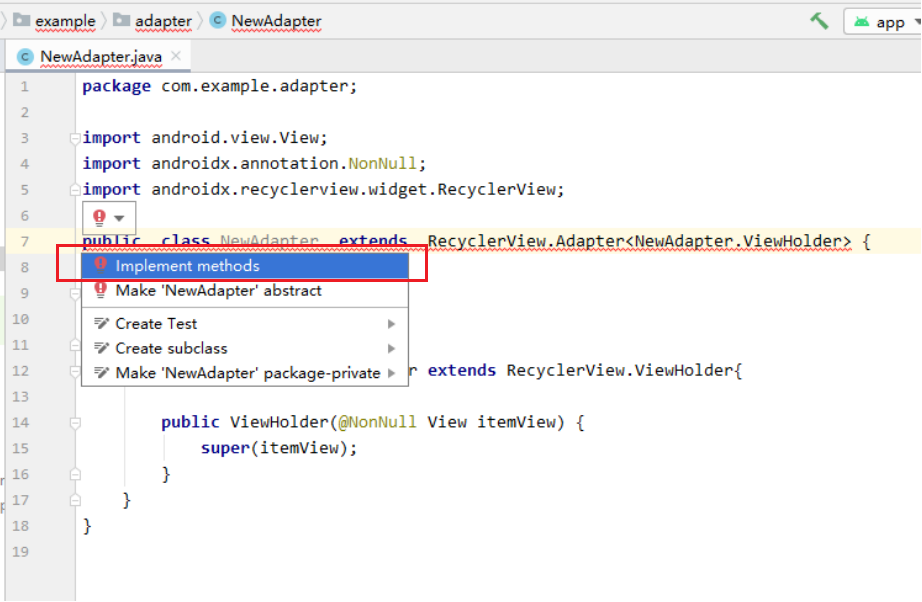
public int getItemCount() {

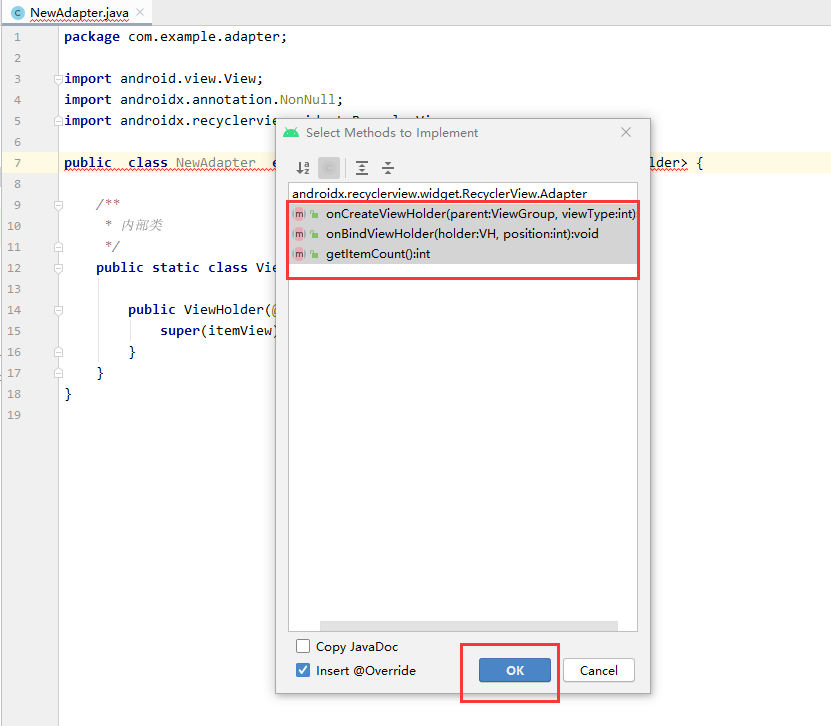
return 0;

}

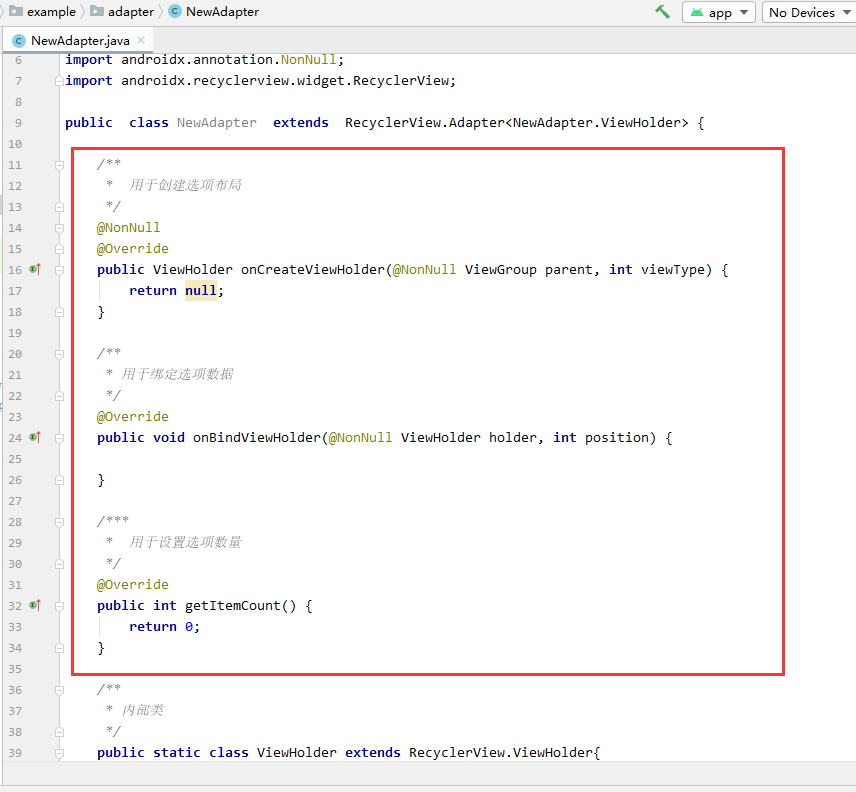
上述代码可以通过便捷方式自动添加，其过程为：鼠标点击代码上报红色波浪线前面部分，稍等一会儿会弹出一个红色的灯泡，点击旁边的选项选择【implements methods】，再在弹出窗口里选定里面的所有方法，再点击【OK】即可。







上述步骤完成后效果如下图所示



其中，onCreateViewHolder方法用于创建选项布局；onBindViewHolder方法用于绑定选项数据； getItemCount方法用于设置选项的数量。上述代码需要对ViewGroup类导包，需要导入

import android.view.ViewGroup;

#### 第13步：定义适配器构造方法

把下面代码插入到适配器类中，其作用是：定义构造方法，用于传入Activity对象和列表选项里的图片和文本数据。

private FragmentActivity activity;

private int[] image;

private String[] text;

public NewAdapter(FragmentActivity activity, int[] image, String[] text){

this.activity=activity;

this.image=image;

this.text=text;

}

上述代码插入的具体位置如下图所示

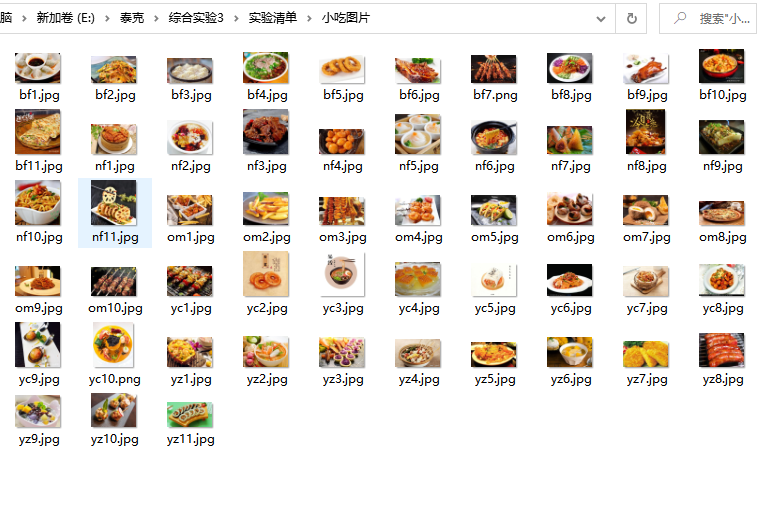


上述代码在编写的时候需要对FragmentActivity类导包，需要导入

import androidx.fragment.app.FragmentActivity;

#### 第14步：把小吃图片复制到项目里

打开【实验清单】里的【小吃图片】文件夹，把里面所有图片复制粘贴到项目中的<app\res\drawable\>目录下，具体步骤参考综合实验1实验步骤第2步，共有53张图片。



#### 第15步：定义initRecyclerView方法

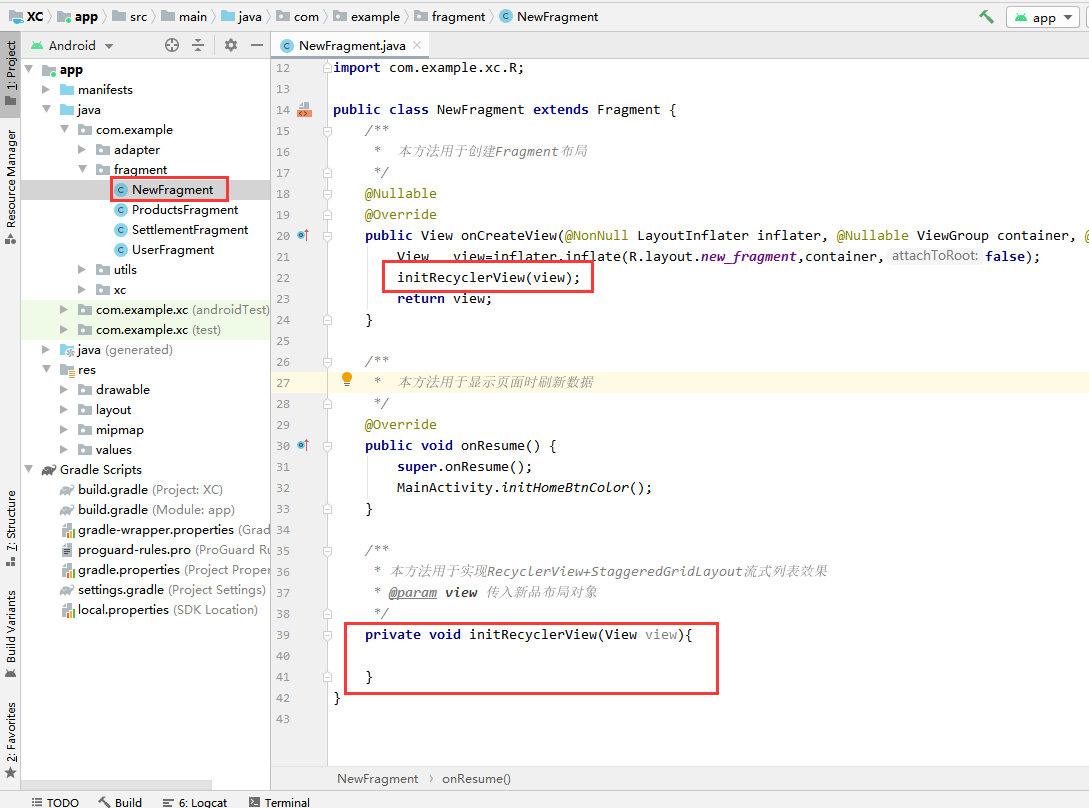
在<app\java\com.example.fragment\>目录下的【NewFragment】类里插入下面两段代码，其作用是：定义并调用initRecyclerView方法，用于编写执行RecyclerView + StaggeredGridLayout步骤代码。

initRecyclerView(view);

private void initRecyclerView(View view){

}

上述代码插入的具体位置如下图所示



其中，方法的参数【View view】用于传入新品布局对象。

**检测条件：**

检测<app\java\com.example.fragment\>路径下的【NewFragment.java】文件里是否有【initRecyclerView】关键字；

#### 第16步：实现RecyclerView + StaggeredGridLayout步骤

在上一步定义的initRecyclerView方法里插入下面的代码，其作用为：实现RecyclerView + StaggeredGridLayout核心步骤。

RecyclerView rv=view.findViewById(R.id.newRecycler); //获得RecyclerView控件对象 //设置为StaggeredGridLayout流式布局

rv.setLayoutManager(new StaggeredGridLayoutManager(2,RecyclerView.VERTICAL)); //定义新品小吃图片数组

int[] image={R.drawable.bf11,R.drawable.bf3,R.drawable.om1,R.drawable.om7,R.drawable.yc1,

R.drawable.nf10,R.drawable.nf8}; //定义新品小吃说明数组

String[] text={"嘎嘣脆香煎饼果子","鲜嫩多汁小笼包","薯条鸡肉丸组合","北欧蜜汁奶粉果",

"什锦串串烤","川味凉面","自贡脆脆兔"};

NewAdapter adapter=new NewAdapter(getActivity(),image,text); //调用适配器

rv.setAdapter(adapter); //设置适配器到RecyclerView

上述代码插入的具体位置如下图所示



上述代码在编写时需要对【RecyclerView】类、【StaggeredGridLayoutManager】类和【NewAdapter】类导包，需要导入

import androidx.recyclerview.widget.RecyclerView;

import androidx.recyclerview.widget.StaggeredGridLayoutManager;

import com.example.adapter.NewAdapter;

#### 第17步：编写适配器类功能逻辑

在<app\java\com.example.adapter\>目录下的【NewAdapter】类里插入或编辑为下面的四段代码，其作用为：适配器类各个流程方法功能逻辑的具体实现。

ViewHolder内部类里需要编辑的代码为

ImageView iv;

TextView tv;

public ViewHolder(@NonNull View itemView) {

super(itemView);

iv=itemView.findViewById(R.id.newImage); //获得列表选项图片控件

tv=itemView.findViewById(R.id.newText); //获得列表选项文本控件

}

onCreateViewHolder方法里需要编辑的代码为

//关联列表选项布局

View view= LayoutInflater.from(parent.getContext()).inflate(R.layout.new\_item,parent,false);

ViewHolder holder=new ViewHolder(view); //调用内部类ViewHolder

return holder;

onBindViewHolder方法里需要编辑的代码为

public void onBindViewHolder(@NonNull final ViewHolder holder, @SuppressLint("RecyclerView") int position) {

//异步加载选项数据

new Thread(new Runnable() {

@Override

public void run() {

activity.runOnUiThread(new Runnable() {

@Override

public void run() {

if((image.length!=0)&&(text.length!=0)){

holder.iv.setImageResource(image[position]);

holder.tv.setText(text[position]);

}

}

});

}

}).start();

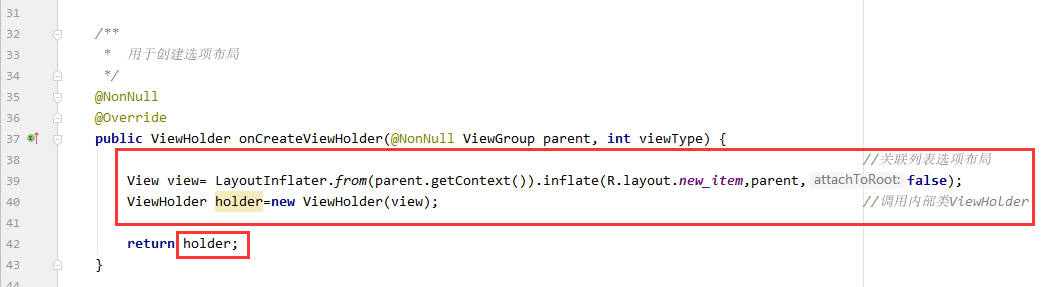
}

getItemCount方法里需要编辑的代码为

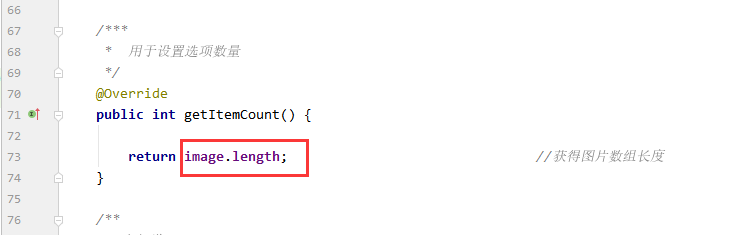
image.length //获得图片数组长度

上述代码插入或编辑的具体位置如下图所示









上述代码在编写时需要导入下面的包

import com.example.xc.R;

import android.widget.ImageView;

import android.widget.TextView;

import android.view.LayoutInflater;

#### 第18步：运行测试

上面所有的步骤都正确实现后，就可以生成apk文件运行测试了。具体步骤参考实验1的第2大步和第3大步，把App安装到手机后的效果如下图所示，可以上下滑动列表区域显示不同新品小吃。