# 综合实验4 《小吃APP》

# 点菜页面设计与左列表功能实现

## 一、实验简介

本次实验将在上次实验代码的基础上实现【点菜页面】设计与其左列表功能逻辑，【点菜页面】分为3部分：标题、左边小吃分类列表、右边小吃信息列表。其主要利用RecyclerView + LinearLayout技术实现普通文本列表效果；

## 二、实验目标

* 熟练RecyclerView列表技术的使用步骤
* 掌握RecyclerView适配器类的LinearLayout的排版布局
* 实现【点菜页面】排版布局

## 实验操作步骤

### 实现效果

### 实现步骤

#### 第1步：打开综合实验3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综合实验3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

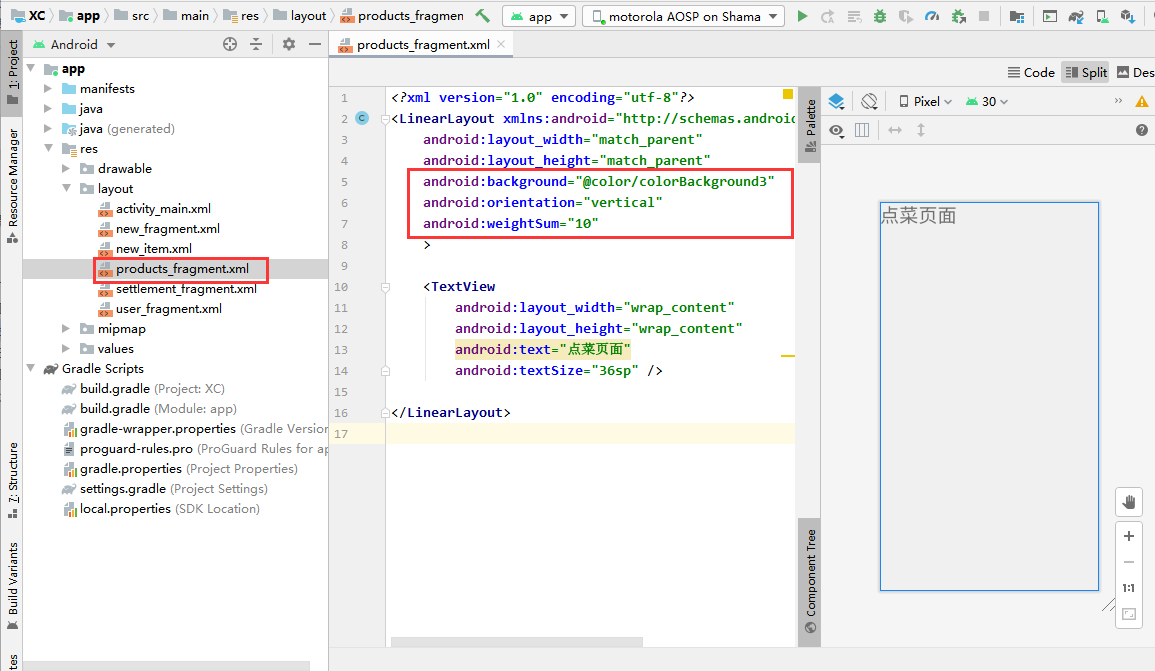
打开<app\res\layout\>目录下的【products\_fragment.xml】布局文件，并插入下面的代码，其效果为：把整个布局分成10等分，添加背景色，改为上下布局结构LinearLayout。

android:background="@color/colorBackground3"

android:orientation="vertical"

android:weightSum="10"

上述代码插入的具体位置如下图所示

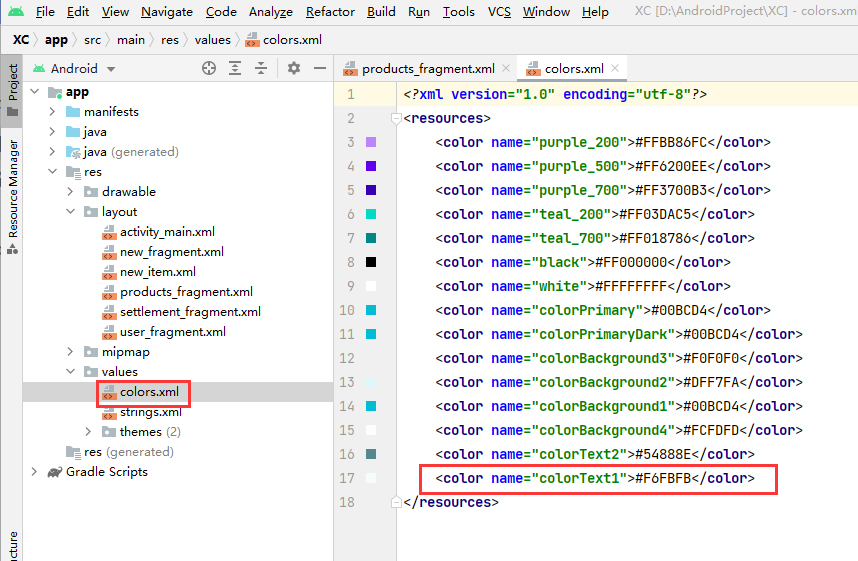


#### 第3步：设计标题

先在<app\res\values\>目录下的【colors.xml】文件里插入下面的代码，其作用为：定义一个颜色。

<color name="colorText1">#F6FBFB</color>

上述代码插入的具体位置如下图所示



再在<app\res\layout\>目录下的【products\_fragment.xml】布局文件里，插入或编辑为下面的代码，其效果为：设计标题【全部小吃】，背景为青蓝色，其控件大小占整个布局1.2等分。

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1.2"

android:background="@color/colorBackground1"

android:gravity="bottom|center"

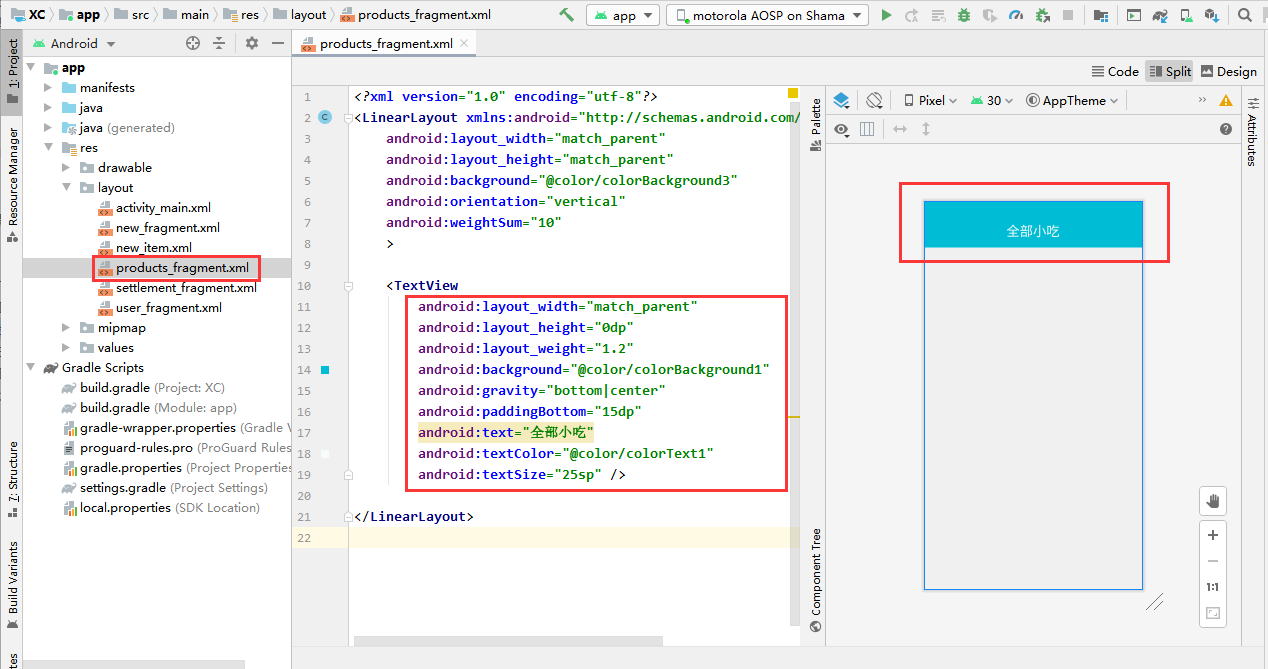
android:paddingBottom="15dp"

android:text="全部小吃"

android:textColor="@color/colorText1"

android:textSize="25sp"

上述代码插入或编辑的具体位置如下图所示



#### 第4步：在布局里添加2个列表控件

在<app\res\layout\>目录下的【products\_fragment.xml】布局文件里插入下面的代码，其效果为：加入2个RecyclerView列表控件到布局里，左右排列，其大小比例为3:7。

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="8.8"

android:orientation="horizontal"

android:weightSum="10">

<androidx.recyclerview.widget.RecyclerView

android:id="@+id/leftRecycler"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_marginTop="7dp"

android:layout\_weight="3" />

<androidx.recyclerview.widget.RecyclerView

android:id="@+id/rightRecycler"

android:layout\_width="0dp"

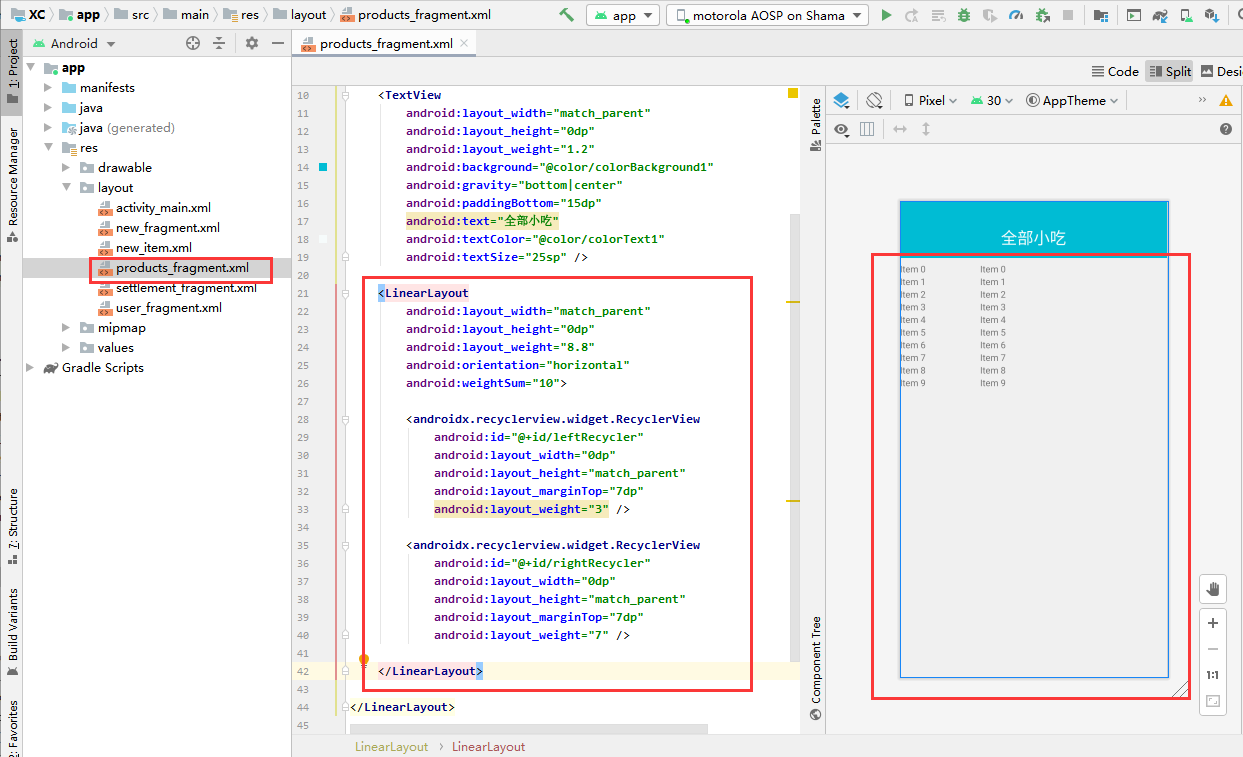
android:layout\_height="match\_parent"

android:layout\_marginTop="7dp"

android:layout\_weight="7" />

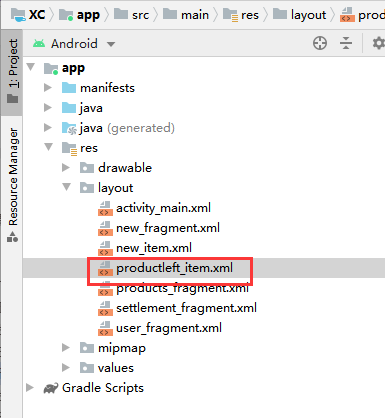
</LinearLayout>

上述代码插入的具体位置和效果如下图所示



#### 第5步：新建左边列表选项布局文件

在<app\res\layout>目录下新建一个名为【productleft\_item.xml】的布局文件，具体新建过程可参考综合实验3实验步骤第6步，完成后如下图所示，其作用为：用于左边列边选项排版布局。



#### 第6步：设计左边列表选项布局

在<app\res\layout\>目录下的【productleft\_item.xml】文件里插入或编辑为下面的代码，其作用是：添加一个TextView控件，高度为50dp，以卡片形式显示，卡片阴影为3dp。

<androidx.cardview.widget.CardView

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

android:layout\_width="match\_parent"

android:layout\_height="50dp"

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

android:orientation="vertical"

app:cardElevation="3dp"

android:layout\_marginRight="5dp"

android:layout\_marginLeft="7dp"

android:layout\_marginBottom="6dp">

<TextView

android:id="@+id/leftText"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_weight="1"

android:gravity="center"

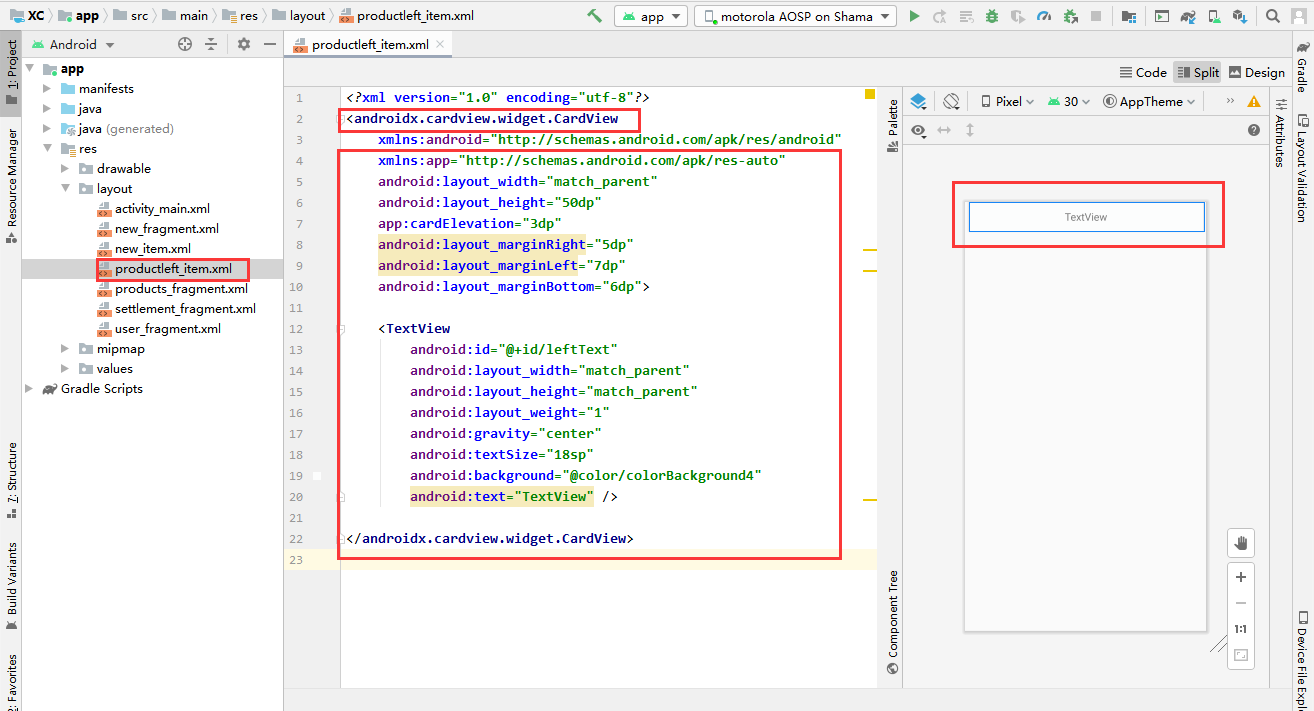
android:textSize="18sp"

android:background="@color/colorBackground4"

android:text="TextView" />

</androidx.cardview.widget.CardView>

上述代码插入或编辑的具体位置和效果如下图 所示



#### 第7步：新建左边列表适配器类

在<app\java\com.example.adapter\>目录下新建一个名为【ProductLeftAdapter】的类，具体新建过程可参考综合实验3实验步骤第8步，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。

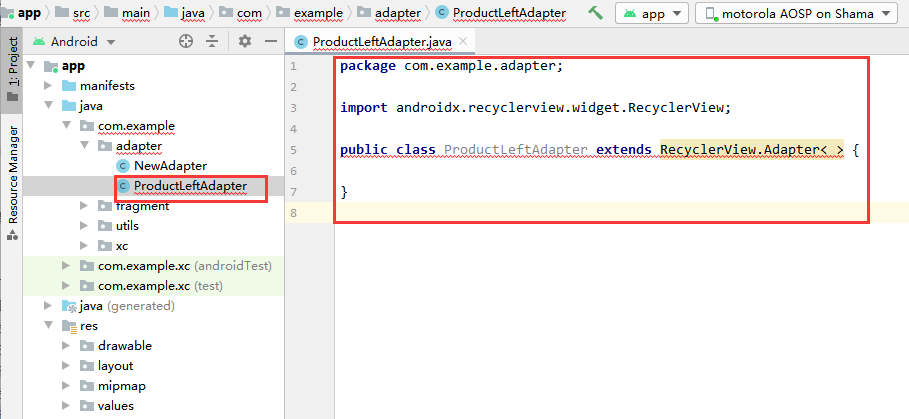
package com.example.adapter;

import androidx.recyclerview.widget.RecyclerView;

public class ProductLeftAdapter 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8步：定义适配器类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。

ProductLeftAdapter.ViewHolder

public static class ViewHolder extends RecyclerView.ViewHolder{

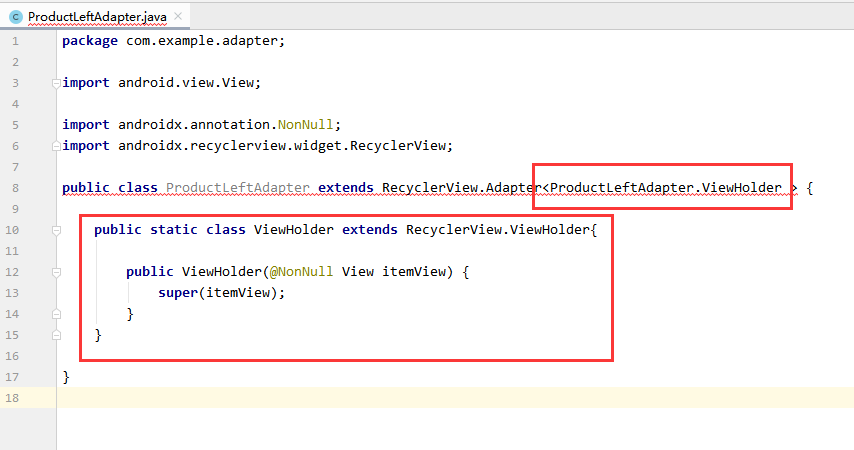
public ViewHolder(@NonNull View itemView) {

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9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) {

}

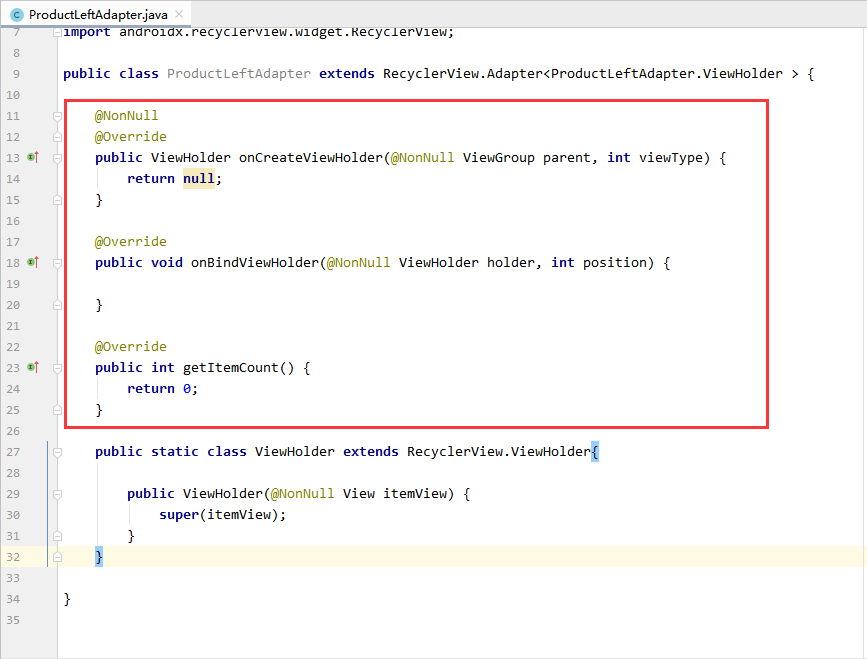
@Override

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，具体可参考综合实验3的实验步骤第12步，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10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小吃分类名。

private FragmentActivity activity;

private String[] text;

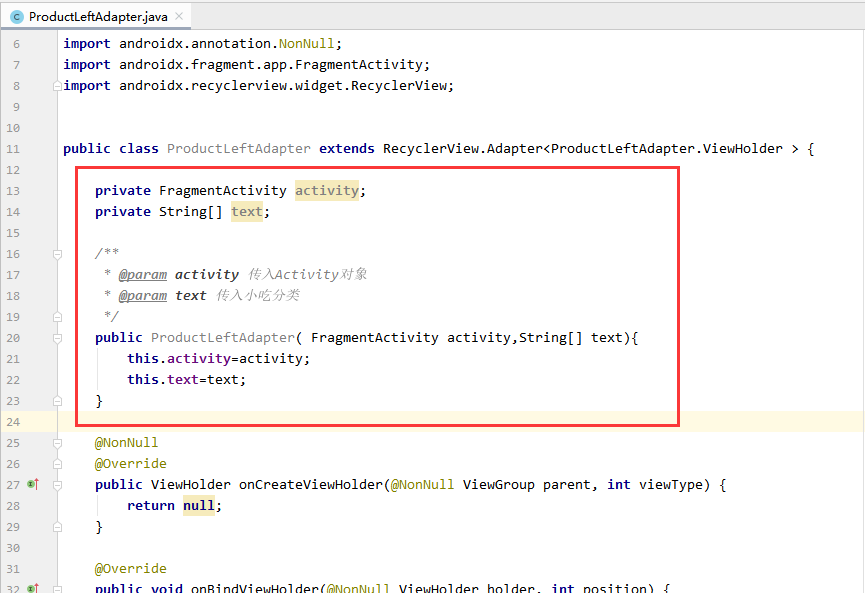
public ProductLeftAdapter( FragmentActivity activity,String[] text){

this.activity=activity;

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11步：定义initRecyclerView方法

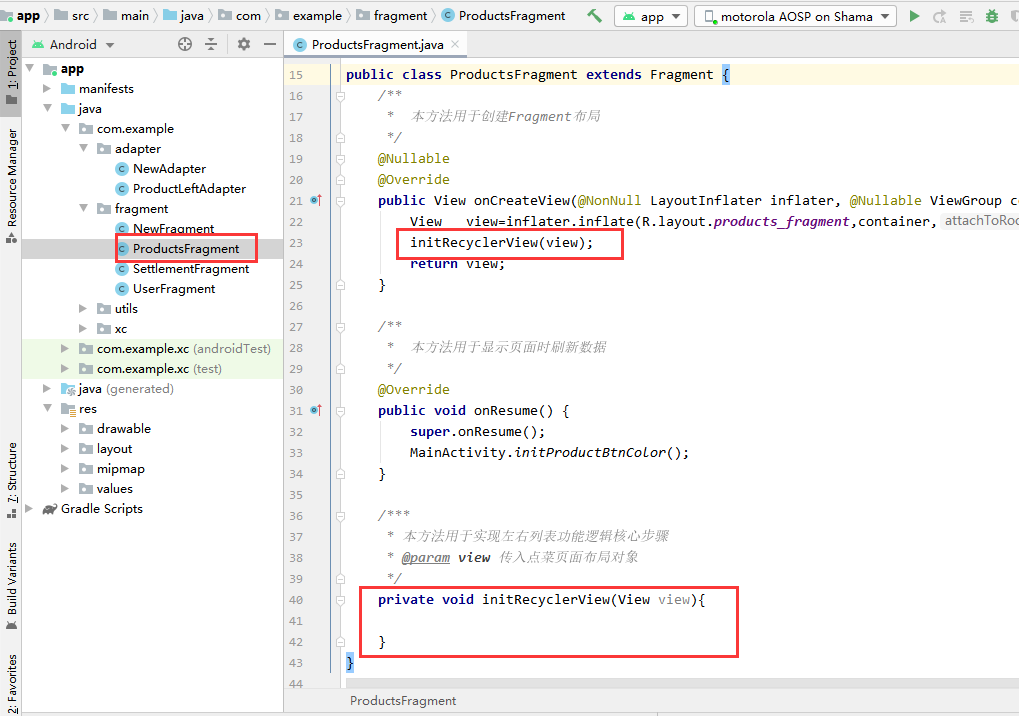
在<app\java\com.example.fragment\>目录下的【ProductsFragment】类里插入下面两段代码，其作用是：定义并调用initRecyclerView方法，用于编写执行左右列表功能逻辑核心步骤代码。

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】用于传入点菜页面布局对象。

#### 第12步：实现RecyclerView + LinearLayout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 + LinearLayout核心步骤。

RecyclerView leftrv=view.findViewById(R.id.leftRecycler); //获得左边列表控件对象

//设置为上下结构的LinearLayout布局

leftrv.setLayoutManager(new LinearLayoutManager(getActivity(),RecyclerView.VERTICAL,false));

//小吃分类名

String[] classname={"南方小吃","北方小吃","原创小吃","亚洲小吃","欧美小吃"};

//调用左边列表适配器类

ProductLeftAdapter leftAdapter=new ProductLeftAdapter(getActivity(),classname);

leftrv.setAdapter(leftAdapter); //把适配器设置给左边列表控件

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、【LinearLayoutManager】类和【ProductLeftAdapter】类导包，需要导入

import androidx.recyclerview.widget.RecyclerView;

import androidx.recyclerview.widget.LinearLayoutManager;

import com.example.adapter.ProductLeftAdapter;

#### 第13步：编写适配器类功能逻辑

在<app\java\com.example.adapter\>目录下的【ProductLeftAdapter】类里插入或编辑为下面的五段代码，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

TextView tv;

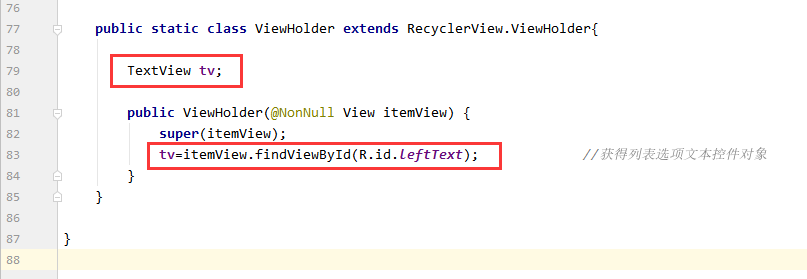
public ViewHolder(@NonNull View itemView) {

super(itemView);

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

}

上述代码插入或编辑的具体位置如下图所示



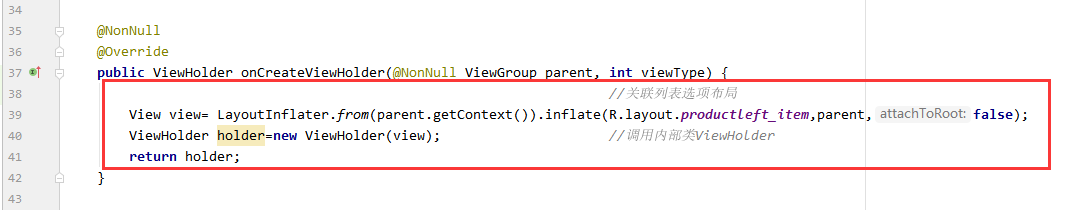
onCreateViewHolder方法里需要编辑的代码为

//关联列表选项布局

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

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

return holder;

上述代码插入或编辑的具体位置如下图所示

onBindViewHolder方法需要编辑的代码为

定义成员变量

private List<ViewHolder> items=new ArrayList<>(); //存放列表所有选项ViewHolder对象

onBindViewHolder方法内部插入

items.add(holder); //添加选项对象到集合

holder.tv.setText(text[position]); //设置选项小吃分类名

//设置默认时的选项颜色

if(position==0){ //默认选中的选项颜色

holder.tv.setTextColor(Color.rgb(84,136,142));

holder.tv.setBackgroundColor(Color.rgb(223,247,250));

}else { //默认没有选中的选项颜色

holder.tv.setTextColor(Color.rgb(148,148,148));

holder.tv.setBackgroundColor(Color.rgb(252,253,253));

}

//点击选项时的操作

holder.tv.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

//没有被点击的选项颜色

for (int i=0;i<items.size();i++){

items.get(i).tv.setTextColor(Color.rgb(148,148,148));

items.get(i).tv.setBackgroundColor(Color.rgb(252,253,253));

}

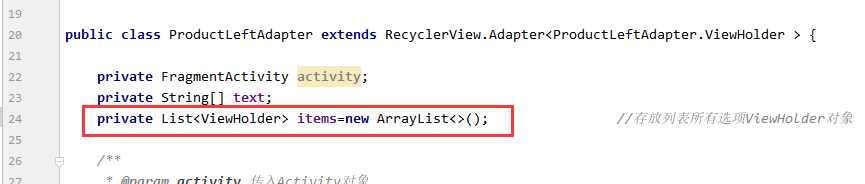
//被点击的选项颜色

holder.tv.setTextColor(Color.rgb(84,136,142));

holder.tv.setBackgroundColor(Color.rgb(223,247,250));

}

});

上述代码插入或编辑的具体位置如下图所示



getItemCount方法里需要编辑的代码为

text.length //获得小吃分类名数组长度

上述代码插入或编辑的具体位置如下图所示



上述几段代码在编写时需要导入下面的包

import com.example.xc.R;

import android.widget.TextView;

import android.view.LayoutInflater;

import java.util.ArrayList;

import java.util.List;

import android.graphics.Color;

#### 第14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，选中选项，其颜色会发生变化。