# 综合实验7 《小吃APP》下单页面功能实现

## 一、实验简介

本次实验将在上次实验代码的基础上实现【下单页面】功能逻辑，没点菜时页面显示【您还没点菜哦】，点过菜时显示所选菜的数据列表及总价。其主要利用RecyclerView + LinearLayout技术实现较为复杂的列表排版效果，利用android:visibility属性实现没点菜和点过菜的界面切换，利用全局类Application作为数据共享媒介；

## 二、实验目标

* 掌握RecyclerView列表里点击事件的处理
* 掌握全局类在页面间共享数据的方式

## 实验操作步骤

### 实现效果

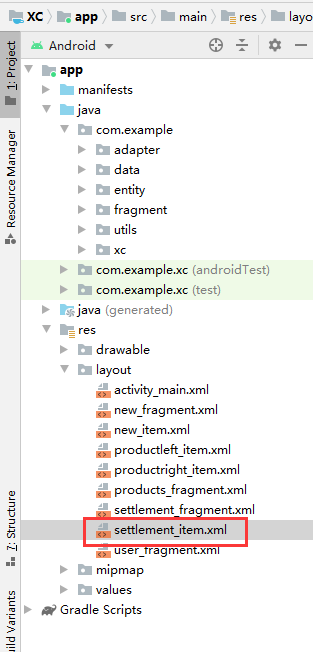
### 实现步骤

#### 第1步：打开综合实验6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6项目【XC】的位置，选择【XC】，再点击【OK】即可。

#### 第2步：新建列表选项布局文件

在<app\res\layout>目录下新建一个名为【settlement\_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：用于已选择的小吃列表选项布局排版。



#### 第3步：把根布局改为CardView控件

在上面新建的布局文件代码里插入或编辑为下面两段代码，其作用为：把选项风格改为卡片效果，选项高度为120dp，卡片四角弧度为8dp，阴影大小为4dp。

androidx.cardview.widget.CardView

android:layout\_height="120dp"

app:cardCornerRadius="8dp"

app:cardElevation="4dp"

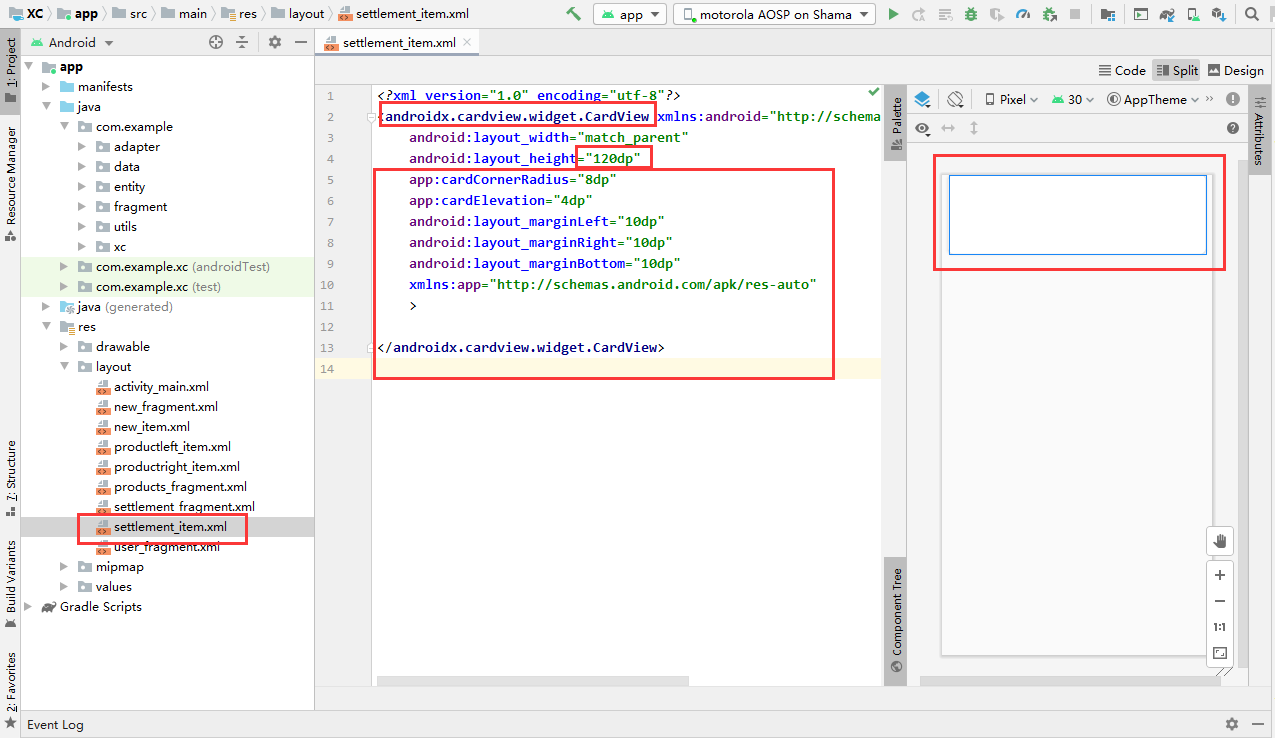
android:layout\_marginLeft="10dp"

android:layout\_marginRight="10dp"

android:layout\_marginBottom="10dp"

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

上述代码插入或编辑的具体位置和效果如下图所示



#### 第4步：设计列表选项整体布局

在上一步代码的基础上插入下面的代码， 其效果是：把选项布局分成左右两块，左边显示小吃图片，大小占整个布局4/10，右边显示小吃其他信息，大小占整个布局6/10。

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@color/colorBackground4"

android:orientation="horizontal"

android:weightSum="10">

<ImageView

android:id="@+id/selectedImage"

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_weight="4"

android:scaleType="centerCrop"

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

<LinearLayout

android:layout\_width="0dp"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:layout\_weight="6"

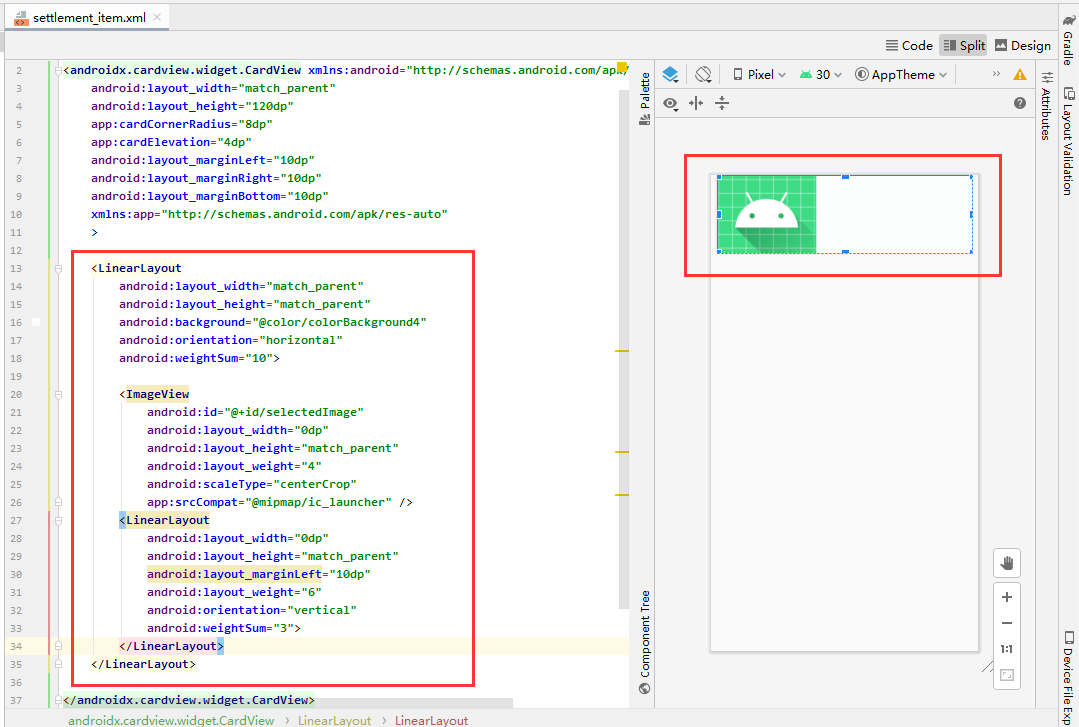
android:orientation="vertical"

android:weightSum="3">

</LinearLayout>

</LinearLayout>

上述代码插入的具体位置和效果如下图所示



#### 第5步：设计列表选项右边整体布局

在上一步代码的LinearLayout(vertical)控件标签里插入下面的代码， 其效果是：把选项右边布局分成上中下三块，上面显示小吃名称，大小占右边布局的1/3；中间显示小吃价格、数量等其他信息，大小占右边布局的1/3；下面可填写小吃备注，大小占右边布局的1/3。

<TextView

android:id="@+id/selectedName"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1"

android:gravity="center|left"

android:text="TextView"

android:textColor="@color/colorText2"

android:textSize="18sp" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1"

android:orientation="horizontal">

</LinearLayout>

<EditText

android:id="@+id/remarksText"

android:layout\_width="match\_parent"

android:layout\_height="0dp"

android:layout\_weight="1"

android:background="#00000000"

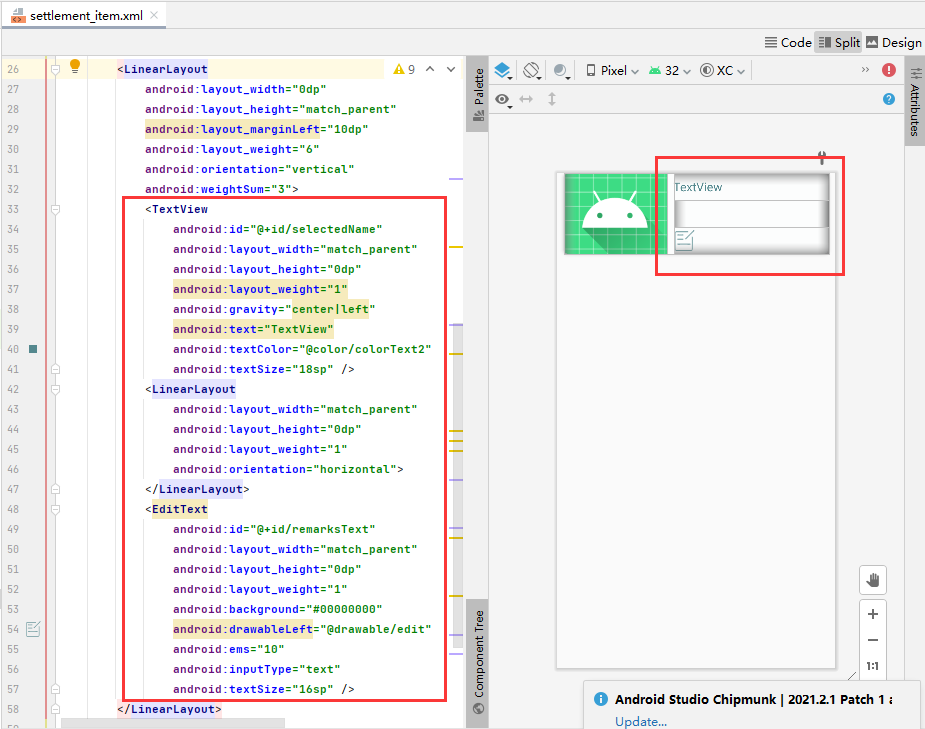
android:drawableLeft="@drawable/edit"

android:ems="10"

android:inputType="text"

android:textSize="16sp" />

上述代码插入的具体位置和效果如下图所示



#### 第5步：设计列表选项右边局部布局

在上一步添加的LinearLayout控件里插入下面的代码，其效果为：添加显示小吃单价控件，添加【+】【-】按钮、小吃数量控件。

<ImageView

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

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

<TextView

android:id="@+id/selectedPrice"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:layout\_marginLeft="5dp"

android:gravity="center|left"

android:text="0.0元/份"

android:textColor="@color/colorText2" />

<Button

android:id="@+id/subtractButton"

android:layout\_width="35dp"

android:layout\_height="35dp"

android:layout\_marginLeft="20dp"

android:background="@color/colorBackground2"

android:text="－"

android:textColor="@color/colorText2"

android:textSize="20sp" />

<TextView

android:id="@+id/selectedNumber"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:layout\_marginRight="10dp"

android:gravity="center"

android:text="1"

android:textColor="@color/colorText2"

android:textSize="18sp" />

<Button

android:id="@+id/addButton"

android:layout\_width="35dp"

android:layout\_height="35dp"

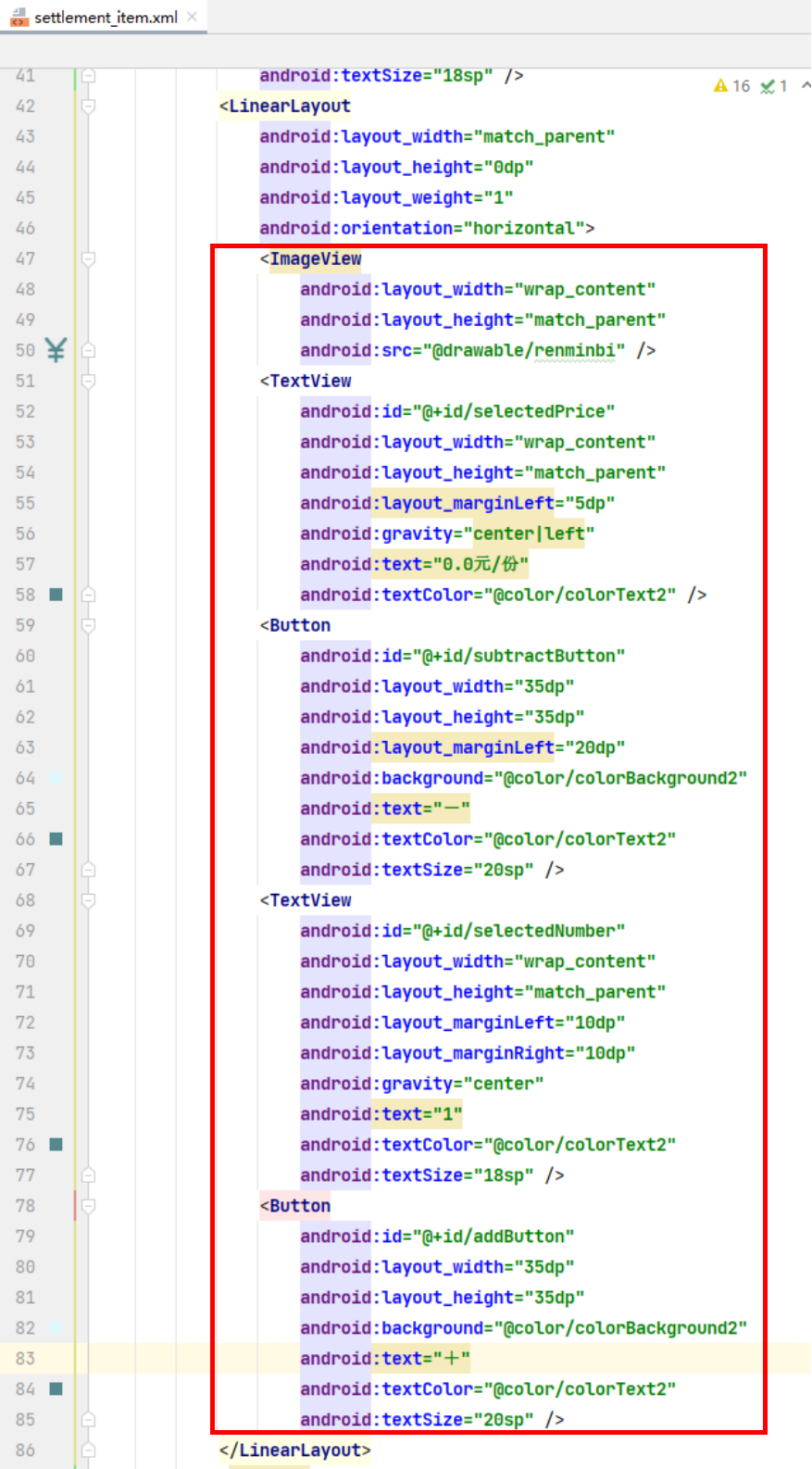
android:background="@color/colorBackground2"

android:text="＋"

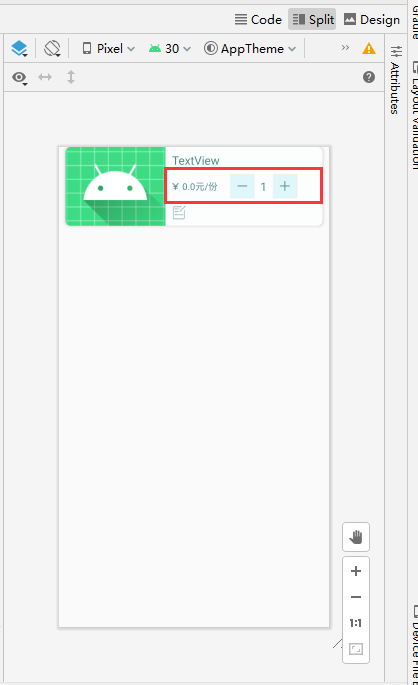
android:textColor="@color/colorText2"

android:textSize="20sp" />

上述代码插入的具体位置如下图所示



上述代码添加完成后的效果如下图所示



#### 第6步：新建右边列表适配器类

在<app\java\com.example.adapter\>目录下新建一个名为【SettlementAdapter】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。

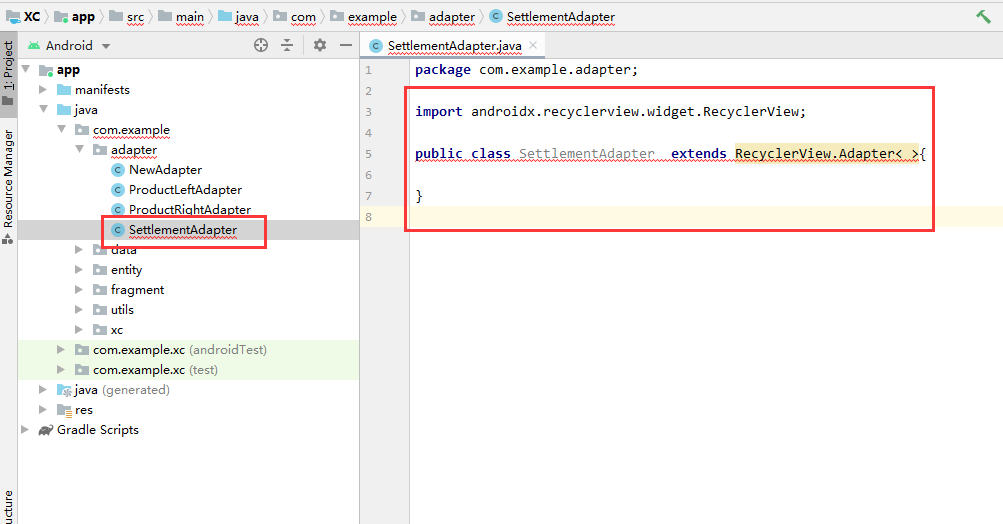
package com.example.adapter;

import androidx.recyclerview.widget.RecyclerView;

public class SettlementAdapter 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7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。

SettlementAdapter.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8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) {

}

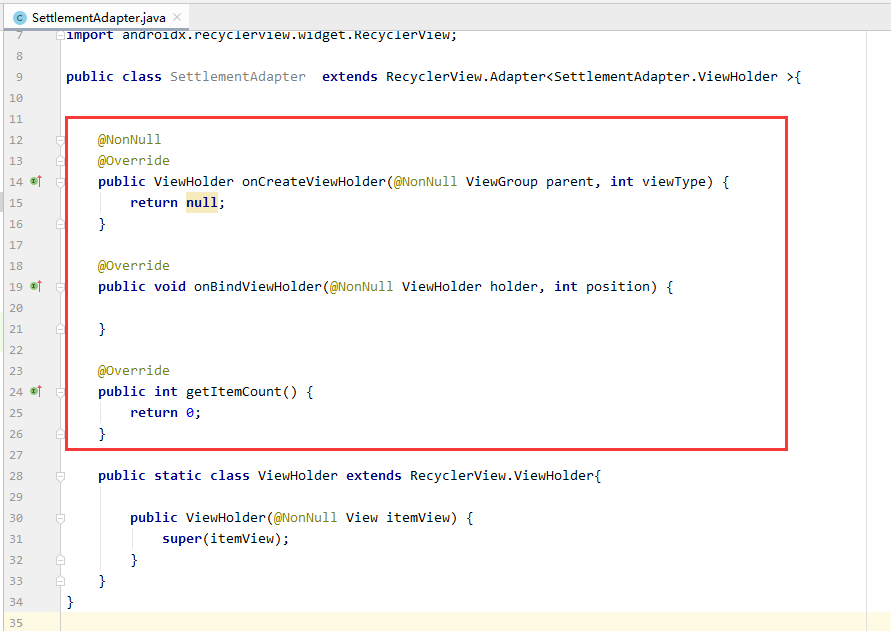
@Override

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9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对象和已选择的小吃数据集合。

private FragmentActivity activity;

private List<Product> list;

public SettlementAdapter(FragmentActivity activity, List<Product> list){

this.activity=activity;

this.list=list;

}

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导入下面的包

import androidx.fragment.app.FragmentActivity;

import com.example.entity.Product;

import java.util.List;

#### 第10步：编写适配器类功能逻辑

在<app\java\com.example.adapter\>目录下的【SettlementAdapter】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

ImageView ivimage;

TextView tvname;

TextView tvprice;

Button btnsubtract;

TextView tvnumber;

Button btnadd;

EditText etremarks;

public ViewHolder(@NonNull View itemView) {

super(itemView);

ivimage=itemView.findViewById(R.id.selectedImage); //获得小吃图片控件对象

tvname=itemView.findViewById(R.id.selectedName); //获得小吃名称控件对象

tvprice=itemView.findViewById(R.id.selectedPrice); //获得小吃价格控件对象

btnsubtract=itemView.findViewById(R.id.subtractButton); //获得【-】按钮控件对象

tvnumber=itemView.findViewById(R.id.selectedNumber); //获得小吃数量控件对象

btnadd=itemView.findViewById(R.id.addButton); //获得【+】按钮控件对象

etremarks=itemView.findViewById(R.id.remarksText); //获得小吃备注对象

}

上述代码插入或编辑的具体位置如下图所示



onCreateViewHolder()方法需要编辑的代码为

定义成员变量如下

private AppData app;

private TextView counttv;

private TextView noproduct;

private ScrollView scrollView;

onCreateViewHolder()方法里插入的代码为

app=(AppData) parent.getContext().getApplicationContext(); //获得全局类

counttv= activity.findViewById(R.id.countPirce); //获得总价控件对象

noproduct=activity.findViewById(R.id.noProduct); //获得没点菜控件对象

scrollView=activity.findViewById(R.id.scrollView); //获得ScrollView控件对象

//关联列表选项布局

View view= LayoutInflater.from(parent.getContext()).inflate(R.layout.settlement\_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方法里插入的代码为

Product product=list.get(position); //获得当前选项数据集合

//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((list.size()!=0)){

holder.ivimage.setImageResource(product.getImage());

holder.tvname.setText(product.getName());

holder.tvnumber.setText(product.getNumber()+""); holder.tvprice.setText(product.getPrice()+"元/份");

}

}

});

}

}).start();

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

list.size() //获得选项数据集合长度

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.Button;

import android.widget.ImageView;

import android.widget.TextView;

import android.widget.EditText;

import android.widget.ScrollView;

import android.view.LayoutInflater;

import com.example.data.AppData;

import android.widget.Toast;

#### 第11步：获得下单页面里的控件对象

在<app\java\com.example.fragment\>目录下的【SettlementFragment】类里插入下面两段代码，其作用为：获得全局类，获得下单页面里的控件对象。

下面的代码作为成员变量

private AppData app;

private RecyclerView rv;

private TextView noproduct;

private ScrollView scrollView;

private TextView countprice;

private EditText countremarks;

private Button btnsettlement;

下面的代码插入到onCreateView()方法里

app= (AppData) getActivity().getApplication(); //获得全局类

rv=view.findViewById(R.id.selectedRecycler); //获得列表控件对象

noproduct=view.findViewById(R.id.noProduct); //获得没点菜时的控件对象

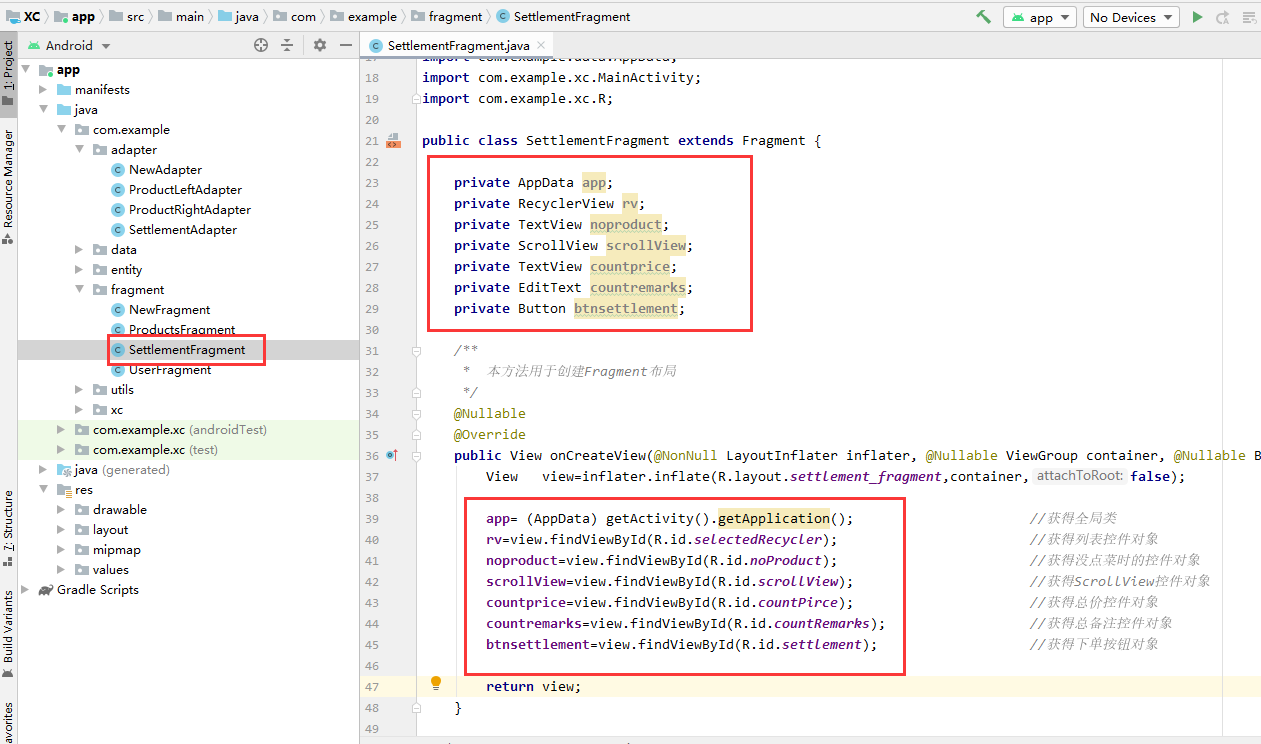
scrollView=view.findViewById(R.id.scrollView); //获得ScrollView控件对象

countprice=view.findViewById(R.id.countPirce); //获得总价控件对象

countremarks=view.findViewById(R.id.countRemarks); //获得总备注控件对象

btnsettlement=view.findViewById(R.id.settlement); //获得下单按钮对象

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导入下面的包

import com.example.data.AppData;

import androidx.recyclerview.widget.RecyclerView;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ScrollView;

import android.widget.TextView;

#### 第12步：实现没点菜和已点菜界面的切换

在<app\java\com.example.fragment\>路径下的【SettlementFragment】类里的onResume方法里插入下面的代码,其作用为：实现没点菜和已点菜界面的切换。

if(app.selectedproduct.size()!=0){ //如果已选小吃集合长度不等于0

noproduct.setVisibility(View.GONE); //就隐藏没点菜控件

scrollView.setVisibility(View.VISIBLE); //就显示已点菜控件

}

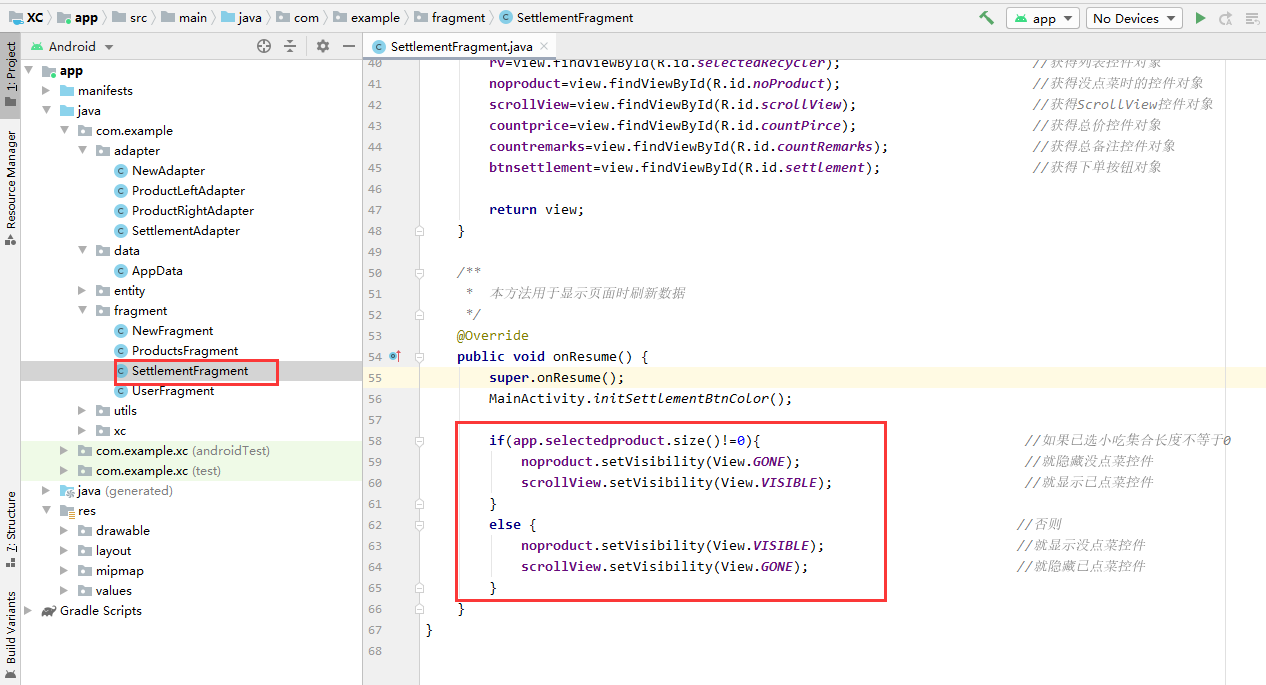
else { //否则

noproduct.setVisibility(View.VISIBLE); //就显示没点菜控件

scrollView.setVisibility(View.GONE); //就隐藏已点菜控件

}

上述代码插入的具体位置如下图所示



#### 第13步：定义initView方法

在<app\java\com.example.fragment\>路径下的【SettlementFragment】类里插入下面两段代码，其作用为：定义initView方法，并调用，该方法用于编写初始化页面数据的代码。

initView();

private void initView(){

}

上述代码插入的具体位置如下图所示

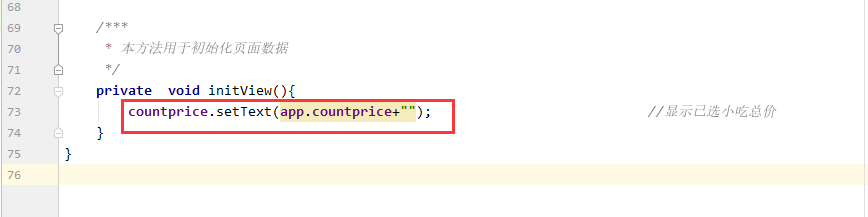


#### 第14步：实现显示总价

在上一步定义的initView方法里插入下面的代码，其效果为：在点菜页面选择小吃后，在下单页面可以同步看到所选小吃的总价。

countprice.setText(app.countprice+""); //显示已选小吃总价

上述代码插入的具体位置如下图所示



#### 第15步：实现RecyclerView + LinearLayout步骤

在上一步代码的基础上添加下面的代码，其作用为：实现RecyclerView + LinearLayout核心步骤。

下面代码插入到initRecyclerView方法里

//设置列表为上下结构的LinearLayout布局

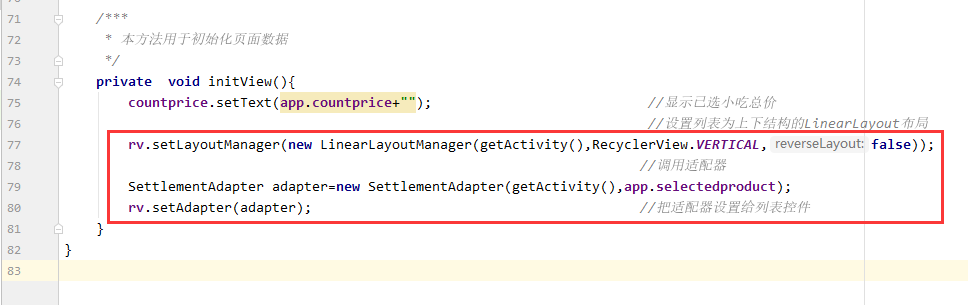
rv.setLayoutManager(new LinearLayoutManager(getActivity(),RecyclerView.VERTICAL,false));

//调用适配器

SettlementAdapter adapter=new SettlementAdapter(getActivity(),app.selectedproduct);

rv.setAdapter(adapter); //把适配器设置给列表控件

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导入下面的包

import androidx.recyclerview.widget.LinearLayoutManager;

import com.example.adapter.SettlementAdapter;

#### 第16步：实现点击【+】【-】按钮相关功能

在<app\java\com.example.adapter\>目录下的【SettlementAdapter】类里的onBindViewHolder方法里，插入或编辑为下面两段代码，其效果为：点击【+】按钮该小吃数量加1，点击【-】按钮该小吃数量减1，且总价也相应调整。

@SuppressLint("RecyclerView")

//点击【+】按钮

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

@Override

public void onClick(View view) {

int num=product.getNumber()+1; //每点一次数量加1

product.setNumber(num); //设置当前数量

holder.tvnumber.setText(num+""); //显示当前数量

//添加当前价格到总价

app.countprice +=Double.parseDouble(product.getPrice()); counttv.setText(app.countprice+""); //显示当前总价

}

});

//点击【-】按钮

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

@Override

public void onClick(View view) {

if(product.getNumber()>1){ //如果数量大于1

int num=product.getNumber()-1; //每点一次数量减1

product.setNumber(num); //设置当前数量

holder.tvnumber.setText(num+""); //显示当前数量

//从总价里减去当前价格

app.countprice -=Double.parseDouble(product.getPrice());

counttv.setText(app.countprice+""); //显示当前总价

}

else if(product.getNumber()==1){ //如果数量等于1

int num=product.getNumber()-1; //每点一次数量减1

product.setNumber(num); //设置当前数量

//从已选小吃集合里删除当前选项数据

app.selectedproduct.remove(position);

//从总价里减去当前价格

app.countprice -=Double.parseDouble(product.getPrice());

counttv.setText(app.countprice+""); //显示当前总价

notifyDataSetChanged(); //刷新列表数据

}

if(app.selectedproduct.size()==0){ //如果已选小吃集合长度等于0

noproduct.setVisibility(View.VISIBLE); //就显示没点菜界面

scrollView.setVisibility(View.GONE); //就隐藏已点菜界面

}

}

});

上述代码插入或编辑的具体位置如下图所示



#### 第17步：实现点击下单按钮

在<app\java\com.example.fragment\>路径下的【SettlementFragment】类里的initView方法里面插入下面的代码，其作用为：点击下单按钮提示“暂时无法下单”。

//点击下单按钮

btnsettlement.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

//提示框提示“暂时无法下单”

Toast.makeText(getActivity(),"暂时无法下单",Toast.LENGTH\_SHORT).show();

}

});

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Toast类导包，需要导入

import android.widget.Toast;

#### 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。

没有点菜时的效果如下图所示



点过菜时的效果如下图所示

点击【+】【-】按钮时的效果如下图所示



点击下单按钮时的效果如下图所示

