中山大学移动信息工程学院本科生实验报告

（2017 年秋季学期）

课程名称：移动应用开发　　任课教师：

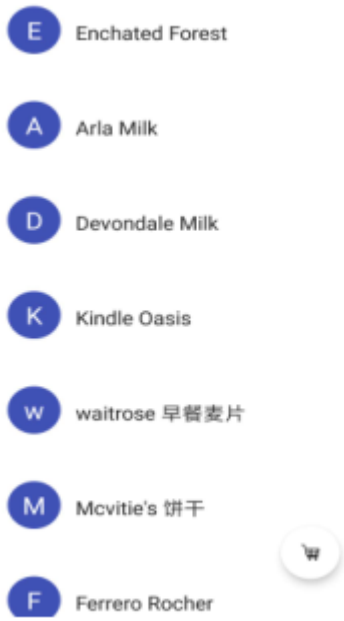
年级	15 级	专业（方向）	软件工程（移动信息工程）
学号	15352223	姓名	刘朝开
电话	15626489095	Email	1209354383@qq.com
开始日期	2017.9.30	完成日期	2017.10.1

一，实验题目

Intent, Bundle 的使用以及 RecyclerView, ListView 的应用

二，实现内容

本次实验模拟实现一个商品表，有两个界面，第一个界面用于呈现商品，如下所示：



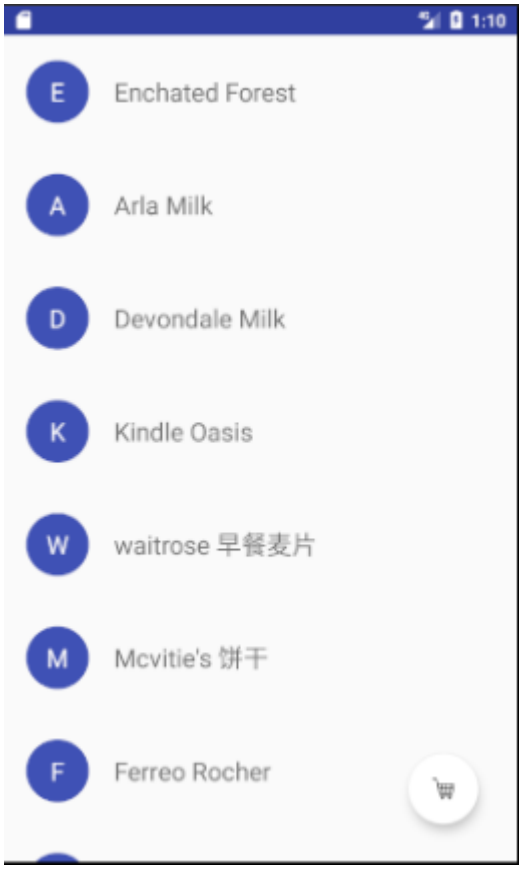
点击右下方的悬浮按钮可以切换到购物车：

上面两个列表点击任意一项后，可以看到详细的信息：

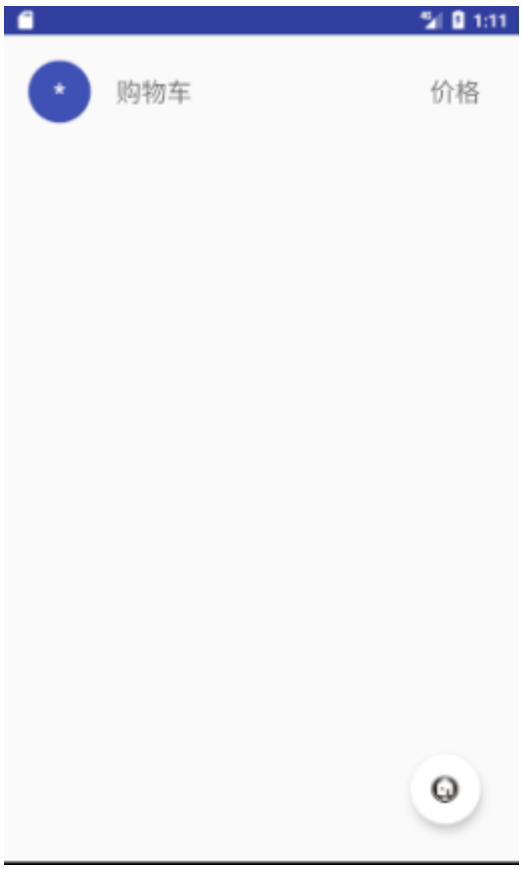
三，课堂实验结果

(1)实验截图

商品列表：


1:10
E
Enchated Forest
A
Arla Milk
D
Devondale Milk
K
Kindle Oasis
W
waitrose 早餐麦片
M
Mcvitie's 饼干
F
Ferreo Rocher

购物车列表：


1:11
购物车
价格

商品详情：

(2)实验步骤以及关键代码

1)首先做出商品列表和购物车列表界面（如下图）：activity_main.xml 文件包括三个控件，recycleview,listview 和 floatactionbutton。Recycleview 用于显示商品列表，listview 用于显示购物车列表，floatactionbutton 用于显示悬浮按钮。Recycleview 和 listview 共用同一个 xml 文件，product.xml。有三个 textview 控件，分别用于显示商品首字母，商品名称，商品价格。

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/shoplist"
    app:backgroundTint="#ffffff"
    android:layout_alignParentBottom="true"
    android:layout_alignParentEnd="true"
    android:layout_margin="30dp"/>

<android.support.v7.widget.RecyclerView
    android:id="@+id/recyclerview"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<ListView
    android:id="@+id/shoppingcar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```

2)新建 Shopping activity 和 activity_shopping.xml，用于显示商品详情界面：我在该布局中设置了三个子布局：两个 relativelayout 和一个 linearlayout,权重分别为 2，1，3。第一个子布局用于显示商品图片，返回按键，商品名字，星星。第二个子布局用于显示价格，信息，加入购物车按钮。第三个子布局用于显示更多产品信息，listview。(布局代码比较简单，不贴图)

3)构建商品类，代码如下。包括三个私有成员和三个成员函数，分别为商品首字母，名字，价格。三个 get()函数。

```
public class Products
{
    private String fletter;
    private String name;
    private String price;

    public Products(String fletter,String name,String price)
    {
        this.fletter = fletter;
        this.name = name;
        this.price = price;
    }
    public  String getFletter(){return fletter;}
    public String getName() {return name;}
    public String getPrice() {return price;}
}
```

4)新建一个 ProductAdapter 类继承于 RecyclerView.Adapter。包括三个私有成员

```
private List<Products> mProductList;
private Context mContext;
private LayoutInflater inflater;
```

类构造函数：

```
public ProductAdapter(Context context,List<Products> productsList)
{
    this.mContext = context;
    this.mProductList = productsList;
    inflater = LayoutInflater.from(mContext);
    //mItemClickListener = null;
}
```

定义内部类继承 RecyclerView.ViewHolder

```
// 定义内部类继承RecyclerView.ViewHolder
static class ViewHolder extends RecyclerView.ViewHolder
{
    TextView pfletter;
    TextView pname;
    //TextView pprice;

    public ViewHolder(View view)
    {
        super(view);
        pfletter = (TextView) view.findViewById(R.id.firstletter);
        pname = (TextView) view.findViewById(R.id.productname);
        //pprice = (TextView) view.findViewById(R.id.price);
    }
}
```

重载函数 ViewHolder onCreateViewHolder 和 void onBindViewHolder 和 int getItemCount(),代码如下所示：

```
@Override
public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType)
    View view = inflater.inflate(R.layout.product,parent,false);
    ViewHolder holder = new ViewHolder(view);
    return holder;
}
@Override
public void onBindViewHolder(ViewHolder holder, final int position)
{
    Products product = mProductList.get(position);
    holder.pfletter.setText(product.getFletter());
    holder.pname.setText(product.getName());
@Override
public int getItemCount() { return mProductList.size(); }
```

5)在 MainActivity 中显示使用 recyclerView 显示布局：

首先定义数组 productsList,布局对象 recyclerView，商品适配器 productAdapter。

```
private List<Products> productsList = new ArrayList<>();
private RecyclerView recyclerView;//获取布局实例
private ProductAdapter productAdapter;
```

第二步，初始化商品列表：

```
initProducts();//初始化产品列表
```

这里我调用了自己的初始化函数：

```
private void initProducts()
{

    for(int i=1;i<=10;i++)
    {
        Products product = new Products(Keywords[i],Names[i],Prices[i]);
        productsList.add(product);
    }
}
```

第三步，赋值获取实例等。代码如下（包括了动画效果）：

```
recyclerView = (RecyclerView) findViewById(R.id.recyclerview);//获取布局实例
productAdapter = new ProductAdapter(MainActivity.this,productsList);
LinearLayoutManager layoutManager = new LinearLayoutManager(this);//设置布局管理器
recyclerView.setLayoutManager(layoutManager); //设置为垂直布局，这也是默认的
//recyclerView.setAdapter(productAdapter);//设置Adapter
ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(productAdapter);
animationAdapter.setDuration(1000);
recyclerView.setAdapter(animationAdapter);
recyclerView.setItemAnimator(new OvershootInLeftAnimator());
```

6)设置 recyclerView 的点击事件。

在 ProductAdapter.java 文件中定义接口类：

```
//定义item的回调接口类
public interface OnItemClickListener
{
    void onItemClick(int position);
    void onItemLongClick(int position);
}
```

新建一个接口类的对象，并在构造函数中置为 null：

```
private OnItemClickListener mItemClickListener;

mItemClickListener = null;
```

定义一个设置点击的监听方法

```
//定义一个设置点击监听的方法
public void setOnItemClickListener(OnItemClickListener onItemClickListener){
    this.mItemClickListener = onItemClickListener;
}
```

在 onBindViewHolder 函数中设置点击事件：

```
if(mItemClickListener != null)//判断是否设置了监听器
{
    holder.itemView.setOnClickListener((v) -> {
            mItemClickListener.onItemClick(position);
    });

    holder.itemView.setOnLongClickListener((v) -> {
            mItemClickListener.onItemLongClick(position);
            return true;
    });
}
```

在 MainActivity 里面设置点击事件：

```
//recycleview点击事件
productAdapter.setOnItemClickListener(new ProductAdapter.OnItemClickListener()
{
    @Override
    //短按跳转至商品详情界面
    public void onItemClick(int position) {
        Intent intent = new Intent(MainActivity.this,Shopping.class);
        final Products product = productsList.get(position);
        intent.putExtra(sName,product.getName());//传入数据
        startActivityForResult(intent,1);//1为请求码，需要是唯一值
    }

    @Override
    //长按删除商品
    public void onItemLongClick(int position) {
        final Products product = productsList.get(position);
        Toast.makeText(MainActivity.this,"移除第"+(position+1)+"个商品",Toast.LENGTH_SHORT).show();
        productsList.remove(product);
        productAdapter.notifyDataSetChanged();
    }
```

7)新建 ShopAdapter 适配器继承于 ArrayAdapter.

成员变量和构造函数如图：

```
private int resourceId;

public ShopAdapter(Context context, int textViewResourceId,
                   List<Products> objects)
{
    super(context,textViewResourceId,objects);
    resourceId = textViewResourceId;
}
```

重载函数 View getView

```
public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
    Products products = getItem(position);
    View view;
    ViewHolder viewHolder;

    if(convertView == null)
    {
        view = LayoutInflater.from(getContext()).inflate(resourceId,parent,false);
        viewHolder = new ViewHolder();
        viewHolder.fLetter = (TextView) view.findViewById(R.id.firstletter);
        viewHolder.pName = (TextView) view.findViewById(R.id.productname);
        viewHolder.pPrice = (TextView) view.findViewById(R.id.price);
        view.setTag(viewHolder);
    }
    else
    {
        view = convertView;
        viewHolder = (ViewHolder) view.getTag();
    }
    viewHolder.fLetter.setText(products.getFletter());
    viewHolder.pName.setText(products.getName());
    viewHolder.pPrice.setText(products.getPrice());
    return view;
```

其中 ViewHolder 为自定义的内部类，用于缓存数据：

```
private class ViewHolder
{
    TextView fLetter;
    TextView pName;
    TextView pPrice;
}
```

8)在 MainActivity 中显示使用 listView 显示布局：

首先定义数组 shopList,布局对象 listView，商品适配器 shopAdapter。

```
private List<Products> shopList = new ArrayList<>()
private ShopAdapter shopAdapter;
private ListView listView;
```

第二步初始化购物车：

```
initShopcar();//初始化购物车
```

其中 initShopcar()为自定义的函数：

```
private void initShopcar()
{
    Products product = new Products(Keywords[0],Names[0],Prices[0]);
    shopList.add(product);
}
```

第三步，赋值获取实例等。

```
//ListView
shopAdapter = new ShopAdapter(MainActivity.this, R.layout.product,shopList);
listView = (ListView) findViewById(R.id.shoppingcar);
listView.setAdapter(shopAdapter);
listView.setVisibility(View.INVISIBLE);//刚开始设置listview不可见
```

9)设置浮动按钮点击事件：

第一步：设置标记 flag=0，表示界面的切换。

第二步：创建 floatingactionbutton 对象，获取实例。

第三步：实现点击切换界面的功能。

```
fab.setOnClickListener((view) → {
        {
            //切换至购物车界面
            if(flag == 0)
            {
                flag = 1;
                //显示购物车界面
                listView.setVisibility(View.VISIBLE);
                recyclerView.setVisibility(View.INVISIBLE);
                fab.setImageResource(R.drawable.mainpage);
            }

            //切换回商品列表
            else
            {
                flag = 0;
                //显示商品列表界面
                listView.setVisibility(View.INVISIBLE);
                recyclerView.setVisibility(View.VISIBLE);
                fab.setImageResource(R.drawable.shoplist);
            }
        }
});
```

10)实现 listview 点击事件：

```
//短按跳转至商品详情界面
listView.setOnItemClickListener((parent, view, position, id) → {
        Products product = shopList.get(position);
        if(position>=1)
        {
            final Products products = shopList.get(position);
            Intent intent = new Intent(MainActivity.this,Shopping.class);
            intent.putExtra(sName,products.getName());
            startActivityForResult(intent,1);
        }
});
```

```
//长按弹出对话框选择是否删除商品
listView.setOnItemLongClickListener((parent, view, position, id) → {
        final Products product = shopList.get(position);
        final AlertDialog.Builder deleteDialog = new AlertDialog.Builder(MainActivity.this);
        deleteDialog.setTitle("移除商品");
        deleteDialog.setMessage("从购物车移除"+product.getName());
        if(position>0)
        {
            deleteDialog.setPositiveButton("确定", (dialog, which) → {
                    Toast.makeText(MainActivity.this,"成功移除商品"+product.getName(),
                            Toast.LENGTH_SHORT).show();
                    shopList.remove(product);
                    shopAdapter.notifyDataSetChanged();
            });

            deleteDialog.setNegativeButton("取消", (dialog, which) → {
                    Toast.makeText(MainActivity.this,"点击了取消",Toast.LENGTH_SHORT).show();
            });
            deleteDialog.show();
        }
        return true;
});
```

11)Shopping.java 的实现包括三个部分：

Back 按钮：直接调用 finish()销毁 activity。

```
Back.setOnClickListener((v) → { finish(); });
```

Star 图案按钮：实现图片的切换

```
Star.setOnClickListener((v) → {
        if(flag == 1)
        {
            Star.setImageResource(R.drawable.empty_star);
            flag = 0;
        }
        else {
            Star.setImageResource(R.drawable.full_star);
            flag = 1;
        }
});
```

sCar 图案按钮：显示信息，并将参数回传

```
sCar.setOnClickListener((v) → {
        Toast.makeText(Shopping.this,"商品已加到购物车",Toast.LENGTH_SHORT).show();
        Intent intent1 = new Intent();
        String Name = Nameview.getText().toString();
        int return_data = SelectNumer(Name);
        intent1.putExtra("data_return",return_data);
        setResult(RESULT_OK,intent1);
});
```

使用 listView 显示更多信息：

```
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<~>(
        Shopping.this,android.R.layout.simple_list_item_1,mInformation);
ListView listView = (ListView) findViewById(R.id.mInformation);
listView.setAdapter(arrayAdapter);
```

其中上述用到的 SelectNumber 函数如下：

```
private int SelectNumer(String sName)
{
    int return_data = 1;
    switch(sName)
    {
        case "Enchated Forest":return_data=1;break;
        case "Arla Milk":return_data=2;break;
        case "Devondale Milk":return_data=3;break;
        case "Kindle Oasis":return_data=4;break;
        case "waitrose 早餐麦片":return_data=5;break;
        case "Mcvitie's 饼干":return_data=6;break;
        case "Ferreo Rocher":return_data=7;break;
        case "Maltesers":return_data=8;break;
        case "Lindt":return_data=9;break;
        case "Borggreve":return_data=10;break;
    }
    return return_data;
}
```

购物车接收返回参数如下：

```
//接受商品详情界面传回的消息，用于更新购物车列表
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode)
    {
        case 1:
            if(resultCode == RESULT_OK)
            {
                int returnedData = data.getIntExtra("data_return",0);
                Log.d(tag,"点击了"+returnedData);
                Products product = new Products(Keywords[returnedData],
                        Names[returnedData],Prices[returnedData]);
                shopList.add(product);//加入购物车数组
                shopAdapter.notifyDataSetChanged();//更新数据
            }
            break;
        default:
    }
}
```

(3)实验遇到困难以及解决思路

困难 1：实现 recyclerview 的点击事件过程中，由于错误地将其放入到悬浮按钮的内部进行实现，导致多次点击和修改点击时间没有反应。
解决思路：将 recyclerview 的点击事件直接放在 onCreat 函数中实现。

困难 2：悬浮按钮在界面出现很多，并且位置较为凌乱，点击时没有反应或者程序直接崩溃。
解决思路：错误的将悬浮按钮放入到 listview 和 recyclerview 的内部界面中，导致每产生一条 item 就有一个悬浮按钮。应将悬浮按钮的位置移到 activity_main.xml 文件中，即与 listview 和 recyclerview 并列摆放。

困难 3：传参错误，错误的将数组的序号传入商品详情界面和购物车中，导致增加购物车商品时加错物品，商品详情界面名字与图片不对应等的效果。
解决思路：用商品名字代替商品序号来进行传参，获得商品名字之后在与数组进行比较然后得出正确的序号，再将商品放入购物车或者在商品详情界面进行显示。

四，课后实验结果

五，实验思考及感想

这次试验花费的时间比较长，遇到的问题比较多，但也通过这次试验，学了比较多的新知识。学习和使用了 activity 的两种启动方式，学会了 activity 之间使用 intent 进行数据的传递和接收。学习和使用了 android 最常用也是最难用的两大控件，listview 和 recyclerview。在这次试验过程中遇到了很多的 bug，

有的让程序执行出错，包括传参错误，点击删除事件时发生错位等，有的则使程序直接崩溃。自己采用了Toast 输出，Log.d()输出等信息还是比较难去发现错误，只能一点一点地通过注释和取消注释来发现问题和解决问题，非常的麻烦。也发现了自己 debug 能力方面的不足，希望在后续实验中能学会断点调试，以提高自己的 debug 能力。

作业要求：

命名要求: 学号_姓名_实验编号，例如 15330000_林 XX_lab1。

实验报告提交格式为 pdf。

实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。