中山大学数据科学与计算机学院软件工程（移动信息工程）本科生实验报告

移动应用开发实验报告

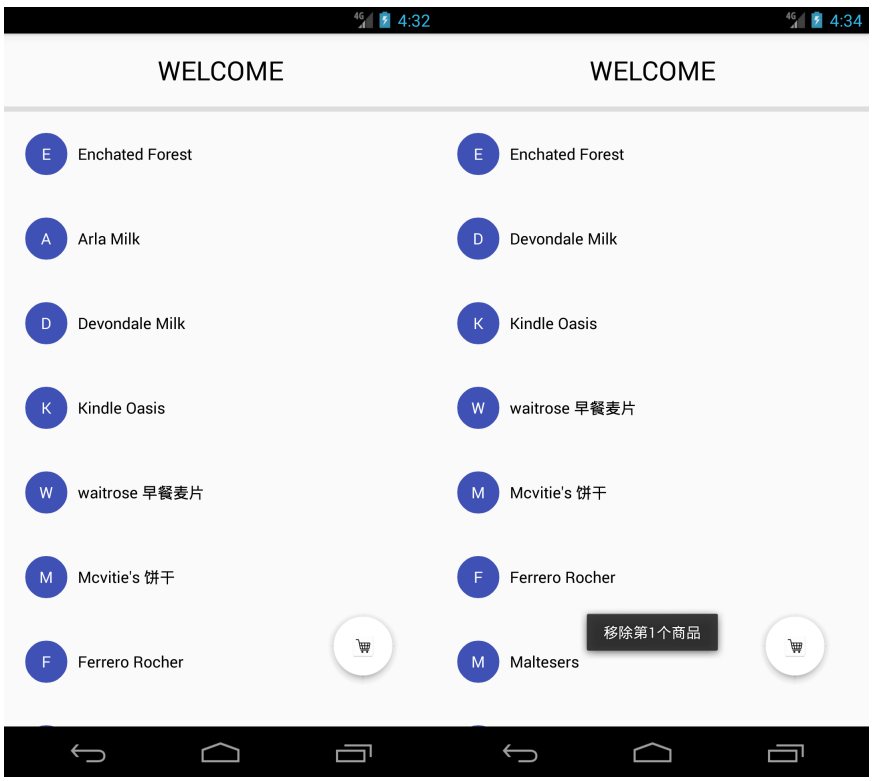
姓名	学号	班级	电话	邮箱	日期
蔡烨	15352010	1501班	15989010669	caiy29@mail2.sysu.edu.cn	2017/11/6

1. 实验题目

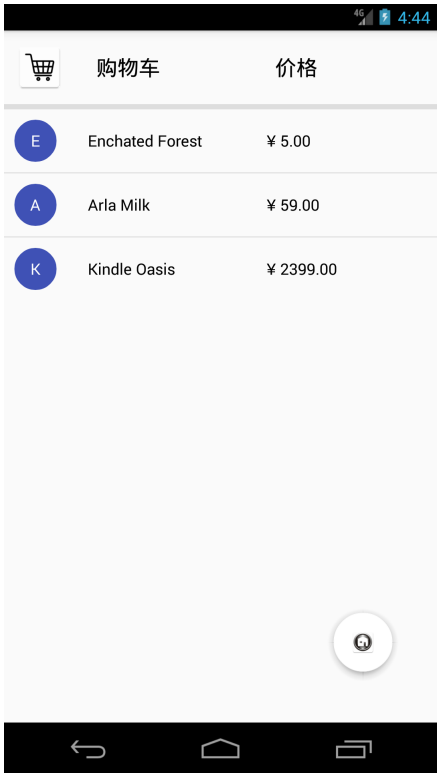
Intent、Bundle的使用以及RecyclerView、ListView的应用

2. 实现内容

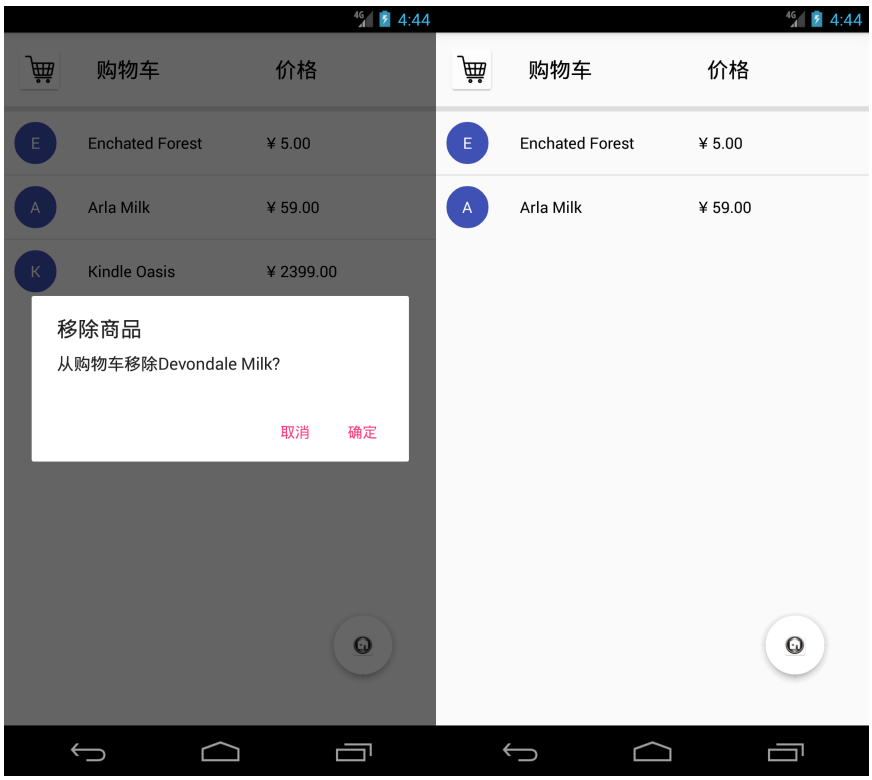
模拟实现一个商品表，有两个节目，第一个界面用于呈现商品，长按商品，可以删除：



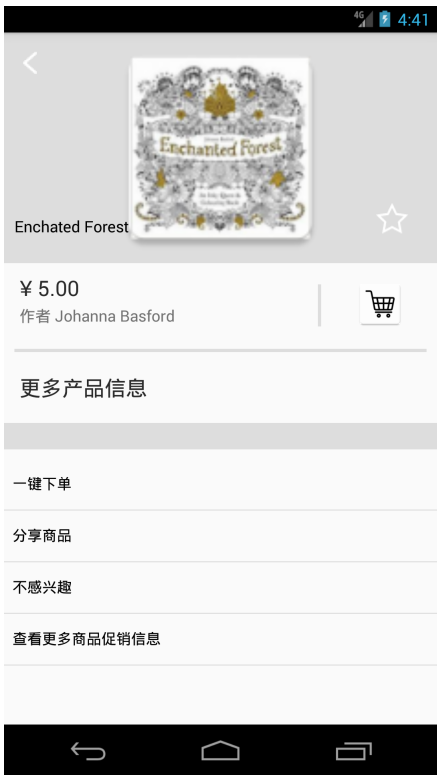
点击下方的悬浮按钮可以切换到购物车，改变按钮图标；



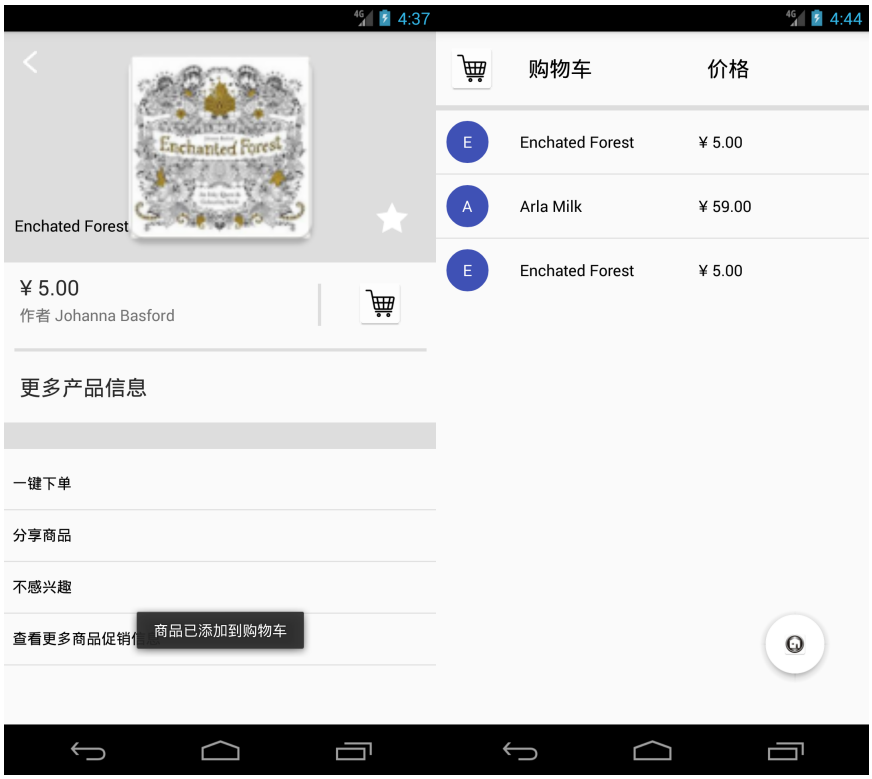
长按商品，弹出对话框，可以选择删除商品：



上面两个列表点击任意一项后，可以看到商品的详细信息：



点击左上角的返回按钮，可以回到原来的界面；点击右边的星星，可以实现空心星星和实心星星的切换；点击购物车按钮，会把商品加到购物车中：



3. 实验过程

1. 将所有需要用到的依赖都添加

```
compile 'com.android.support:cardview-v7:25.3.1'
compile 'com.android.support:recyclerview-v7:25.3.1'
compile 'com.android.support:design:25.3.1'
compile 'jp.wasabeef:recyclerview-animators:2.2.7'
compile 'com.android.support:support-core-utils:25.3.1'
```

```
allprojects {
    repositories {
        jcenter()
        maven {
            url "https://maven.google.com"
        }
    }
}
```

2. 写三个布局文件，实现商品列表和购物车列表界面

   activity_main.xml：

   分别有两个LinearLayout，第一个用来给商品列表布局，里面有RecyclerView；第二个是购物车列表的布局，里面有ListView。之后是一个FloatingActionButton，页面右下方的悬浮按钮。

```
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="WELCOME"
        android:textSize="25dp"
        android:textColor="@color/black"
        android:layout_marginTop="18dp"
        android:layout_marginBottom="18dp"/>
    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="5dp"
        android:background="@color/line"/>
    <android.support.v7.widget.RecyclerView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/recycler_view"/>
</LinearLayout>
```

```
    <ImageView
        android:layout_width="fill_parent"
        android:layout_height="5dp"
        android:background="@color/line"/>
    <ListView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/list_view"></ListView>
</LinearLayout>
<android.support.design.widget.FloatingActionButton
    android:id="@+id/float_car"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@mipmap/shoplist"
    app:backgroundTint="@color/white"
    app:rippleColor="@color/white"
    app:fabSize="normal"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintBottom_toBottomOf="parent"
    android:layout_marginRight="30dp"
    android:layout_marginBottom="30dp" />
</android.support.constraint.ConstraintLayout>
```

   shopcar.xml：购物车列表中每一项商品的布局模板，也就是在填充ListView的时候，用的是这个布局。具体代码见代码文件。

   shoplist.xml：商品列表中每一项商品的布局模板，即在填充RecyclerView时用的布局，具体代码见代码文件。

3. 构造一个商品类，包括每一个商品的名字、价格、信息、图片。

```
public class Good {
    private String name;
    private String price;
    private String info;
    private int pic;

    public  Good(String name,String price,String info,int pic){
        this.name = name;
        this.info = info;
        this.price = price;
        this.pic = pic;
    }

    String getName() { return this.name; }
    String getPrice() { return this.price; }
    String getInfo() {return this.info; }
    int getPic() { return this.pic;}
}
```

4. 为购物车列表的ListView写一个适配器：myListViewAdapter：

   除了重写一些需要的函数外，关键在于自定义类ViewHolder和重写getView函数。ViewHolder可以看成是一个容器，将购物车列表中一个商品item中所有的控件都放进来。当getView函数中，需要判断如果view为空，则在布局上加载listView的模板，将其中的控件对应到viewHolder中的每个控件；如果view不为空，则直接拿到它里面的viewHolder。之后购物车商品的数据赋给viewHolder里对应的控件，完成了对购物车列表数据的加载。

```
public class ViewHolder {
    public TextView cir;
    public  TextView name;
    public  TextView price;
}
@Override
public View getView(int position, View view, ViewGroup parent) {
    View convertView;//新声明一个View变量和ViewHolder变量
    ViewHolder viewHolder;
    if(view == null){//当view为空时才加载布局，并且创建一个viewHolder，获得布局中的三个控件
        //返回inflate的方法加载布局，context这个参数需要使用这个adapter的activity传入
        convertView = LayoutInflater.from(context).inflate(R.layout.shopcar,null);
        viewHolder = new ViewHolder();
        viewHolder.cir = (TextView)convertView.findViewById(R.id.cir_list);
        viewHolder.name = (TextView) convertView.findViewById(R.id.name_list);
        viewHolder.price = (TextView) convertView.findViewById(R.id.price_list);
        convertView.setTag(viewHolder);//将处理好的viewHolder放入item中
    } else {//否则，让convertView等于view，然后从中取出viewHolder即可
        convertView = view;
        viewHolder = (ViewHolder) convertView.getTag();
    }
    //从viewHolder中取出对应的对象，然后赋值给他们
    viewHolder.cir.setText(good.get(position).getName().substring(0,1).toUpperCase());
    viewHolder.name.setText(good.get(position).getName());
    viewHolder.price .setText(good.get(position).getPrice());
    return convertView;
}
```

5. 为商品列表的recyclerView写一个适配器：myRecyclerView：

   除了重写一些常规的函数，recyclerView里没有点击事件的监控器，因此需要在adapter里设置一个监听器，当itemView被点击的时候，调用该监听器并且将itemView的position作为参数传递出去。首先要添加接口：

```
//添加接口和点击函数
public interface OnItemClickListener {
    void onClick(int position);
    void onLongClick(int position);
}
public void setOnItemClickListener(OnItemClickListener onItemClickListener) {
    this.mOnItemClickListener = onItemClickListener;
}
```

自定义viewHolder：

```
class ViewHolder extends RecyclerView.ViewHolder {
    TextView tx1;
    TextView tx2;
    public ViewHolder(View view) {
        super(view);
        tx1 = (TextView) view.findViewById(R.id.cir_rec);
        tx2 = (TextView) view.findViewById(R.id.itemName_rec);
    }
}
```

重写两个函数，在里面要写上点击事情的监听器要做的事，以及对item的数据填充：

```
//返回一个自定义的ViewHolder
@Override
public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.shoplist,parent,false);
    ViewHolder holder = new ViewHolder(view);
    return holder;
}
//填充onCreateViewHolder方法返回的holder中的控件
@Override
public void onBindViewHolder(final ViewHolder holder, int position) {
    holder.tx1.setText(good.get(position).getName().substring(0,1).toUpperCase());
    holder.tx2.setText(good.get(position).getName());

    if(mOnItemClickListener != null){
        holder.itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mOnItemClickListener.onClick(holder.getAdapterPosition());
            }
        });
        holder.itemView.setOnLongClickListener(new View.OnLongClickListener() {
            @Override
            public boolean onLongClick(View v) {
                mOnItemClickListener.onLongClick(holder.getAdapterPosition());
                return false;
            }
```

6. 实现MainActivity.java

从Activity_main.xml中分别获取商品列表的layout和购物车列表的layout，分别为他们设置adapter，同时为recyclerView设置带有动画效果的adapter。

```
myListView = (ListView)findViewById(R.id.list_view);
myRecyclerView = (RecyclerView)findViewById(R.id.recycler_view);
listViewAdapter = new myListViewAdapter(MainActivity.this,carGoods);
recyclerAdapter = new myRecyclerAdapter(MainActivity.this,itemGoods);
myListView.setAdapter(listViewAdapter);
myRecyclerView.setLayoutManager(new LinearLayoutManager(this));
//myRecyclerView.setAdapter(recyclerAdapter);
//有动画的适配器
ScaleInAnimationAdapter animationAdapter = new ScaleInAnimationAdapter(recyclerAdapter);
animationAdapter.setDuration(700);
myRecyclerView.setAdapter(animationAdapter);
myRecyclerView.setItemAnimator(new OvershootInLeftAnimator());
```

将所有的商品信息封装成一个个的Good类，商品列表的商品和购物车的商品分别放到两个List<Good>中。

```
final String[] name = new String[]{"Enchated Forest", "Arla Milk", "Devondale Milk", "Kindl
final String[] price = new String[]{"¥ 5.00", "¥ 59.00", "¥ 79.00", "¥ 2399.00", "¥ 179.00"
final String[] info = new String[]{"作者 Johanna Basford", "产地 德国", "产地 澳大利亚", "版
final int[] pic = new int[]{R.mipmap.enchatedforest,R.mipmap.arla,R.mipmap.devondale,R.mipmap
for(int i=0; i<name.length; i++)
    itemGoods.add(new Good(name[i],price[i],info[i],pic[i]));
recyclerAdapter.notifyDataSetChanged();
```

设置购物车列表的点击事件：点击时进入商品详情页面，发送requestCode=0；长按时弹出对话框，如果选择确定，可以删除该商品：

```
myListView.setOnItemClickListener((parent, view, position, id) -> {
        Intent intent = new Intent(MainActivity.this,DetailsActivity.class);
        intent.putExtra("name",carGoods.get(position).getName());
        intent.putExtra("price",carGoods.get(position).getPrice());
        intent.putExtra("info",carGoods.get(position).getInfo());
        intent.putExtra("pic",carGoods.get(position).getPic());
        startActivityForResult(intent,0);//requestcode=0
});
myListView.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> parent, View view, final int position, long id) {
        AlertDialog.Builder dialog = new AlertDialog.Builder(MainActivity.this);
        dialog.setTitle("移除商品").setMessage("从购物车移除"+name[position]+"?")
                .setNegativeButton("取消", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        Toast.makeText(getApplicationContext(),"您选择了取消",Toast.LENGTH_SHORT).show();
                    }
                })
                .setPositiveButton("确定", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        carGoods.remove(position);
                        listViewAdapter.notifyDataSetChanged();
                    }
                }).show();
        return false;
```

设置商品列表的点击事情，点击商品会进入商品详情页面，发送requestCode=0；长按商品会移除商品，使用带有动画效果的移除：

```
//商品列表的点击事件
recyclerAdapter.setOnItemClickListener(new myRecyclerAdapter.OnItemClickListener() {
    @Override
    public void onClick(int position) {
        Intent intent = new Intent(MainActivity.this,DetailsActivity.class);
        intent.putExtra("name",itemGoods.get(position).getName());
        intent.putExtra("price",itemGoods.get(position).getPrice());
        intent.putExtra("info",itemGoods.get(position).getInfo());
        intent.putExtra("pic",itemGoods.get(position).getPic());
        intent.putExtra("pos",position);
        startActivityForResult(intent,0);
    }

    @Override
    public void onLongClick(int position) {
        Toast.makeText(getApplicationContext(),"移除第"+position+"个商品",Toast.LENGTH_SHORT).show();
        itemGoods.remove(position);
        recyclerAdapter.notifyItemRemoved(position);//有动画的删除
    }
```

设置悬浮图标的点击事件，用setVisibility实现购物车列表和商品列表界面的转换：

```
//商品列表和购物车列表的跳转
final LinearLayout carLayout = (LinearLayout)findViewById(R.id.car_layout);
final LinearLayout homeLayout = (LinearLayout)findViewById(R.id.home_layout);
carLayout.setVisibility(View.INVISIBLE);//购物车列表一开始设置为不可见
final FloatingActionButton floatingCar = (FloatingActionButton)findViewById(R.id.float_car);
floatingCar.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(carLayout.getVisibility()== View.INVISIBLE){
            carLayout.setVisibility(View.VISIBLE);
            homeLayout.setVisibility(View.INVISIBLE);
            floatingCar.setImageResource(R.mipmap.mainpage);
        }
        else {
            carLayout.setVisibility(View.INVISIBLE);
            homeLayout.setVisibility(View.VISIBLE);
            floatingCar.setImageResource(R.mipmap.shoplist);
        }
```

重写onActivityResult函数，如果接收到的requestCode和resultCode都是0，则将商品信息添加到代表购物车商品的List<Good>里。在进入商品详情页面的时候，已经发送了requestCode=0，因此需要在加入购物车的按钮被点击的时候，也发送resultCode=0（DetailsActivity.java中实现）：

```
//更新购物车的数据
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent intent){
    if(requestCode==0 && resultCode==0){
        Bundle bud = intent.getExtras();
        String name = bud.getString("name");
        String price = bud.getString("price");
        String info = bud.getString("info");
        int pic = bud.getInt("pic");
        int count = bud.getInt("cnt",0);
        for(int i=0; i<count; i++)
            carGoods.add(new Good(name,price,info,pic));
        listViewAdapter.notifyDataSetChanged();
    }
```

7. 写一个商品详情页面的布局文件detail.xml，按照顺序一个个写上控件，最后写上一个listView。再写一个moreinfo.xml文件，为listView部分的item写一个模板布局。
8. 写一个商品详情的activity：DetailsActivity.java

   首先，这个activity关联的布局是detail.xml

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate (savedInstanceState);
    setContentView(R.layout.detail);
```

   然后写一个ArrayAdapter，把布局中的listView填充好数据：

```
String [] selection = {"一键下单","分享商品","不感兴趣","查看更多商品促销信息"};
ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(DetailsActivity.this,R.layout.support_simple_spinner_dropdown_item,selection);
listView = (ListView)findViewById(R.id.more);
listView.setAdapter(arrayAdapter);
```

   然后获取从上一个activity中传过来的intent的内容，将对应的内容设置到对应的控件中：

```
intent = getIntent();
Bundle bundle = intent.getExtras();
final String showName = intent.getStringExtra("name");
final String showPrice = intent.getStringExtra("price");
final String showInfo = intent.getStringExtra("info");
final int showPic = bundle.getInt("pic");

name = (TextView)findViewById(R.id.detailName);
price = (TextView)findViewById(R.id.detailPrice);
info = (TextView)findViewById(R.id.detailInfo);
star = (ImageButton)findViewById(R.id.detailStar);
back = (ImageButton)findViewById(R.id.detailBack);
car = (ImageButton)findViewById(R.id.detailCar);
pic = (ImageView)findViewById(R.id.detailPic);
//商品信息更新
name.setText(showName);
price.setText(showPrice);
info.setText(showInfo);
pic.setImageResource(showPic);
```

   然后设置三个监听器，分别对返回按钮、星星、购物车按钮的点击事件进行监听。如果返回按钮被点击，则结束当前的activity，返回到上一个activity；如果星星按钮被监听，则判断当前星星的状态是空心的还是实心的，改成另外的状态，通过setTag来实现：

```
//返回按钮的监听
back.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
//星星按钮的监听
star.setTag("0");
star.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(star.getTag() == "0"){
            star.setBackgroundResource(R.mipmap.full_star);
            star.setTag("1");
        } else {
            star.setBackgroundResource(R.mipmap.empty_star);
            star.setTag("0");
        }
```

如果购物车按钮被点击，则将当前商品的信息都放到intent中，由setResult函数将该intent和resultCode=0传到MainActivity中：

```
//购物车按钮的监听
car.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(DetailsActivity.this,"商品已添加到购物车",Toast.LENGTH_SHORT).show();
        cnt++;
        Intent newIntent = new Intent();
        newIntent.putExtra("cnt",cnt);
        newIntent.putExtra("name",showName);
        newIntent.putExtra("price",showPrice);
        newIntent.putExtra("info",showInfo);
        newIntent.putExtra("pic",showPic);
        setResult(0,newIntent);//resultcode=0
    }
```

至此，整个APP已经完成。

4. 创新点

1. 优化了商品列表的界面
2. 移除商品列表的商品时，加入了动画

5. 实验思考及感想

这次实验是前几次中难度最大的一次，除了内容多外，更多的是刚接触的知识点，光靠TA的实验文档和老师课上的PPT，无法真正的理解和使用，还要经常靠搜索引擎看看别人的博客、代码，以及请教同学，才能真正的学会如何运用知识。