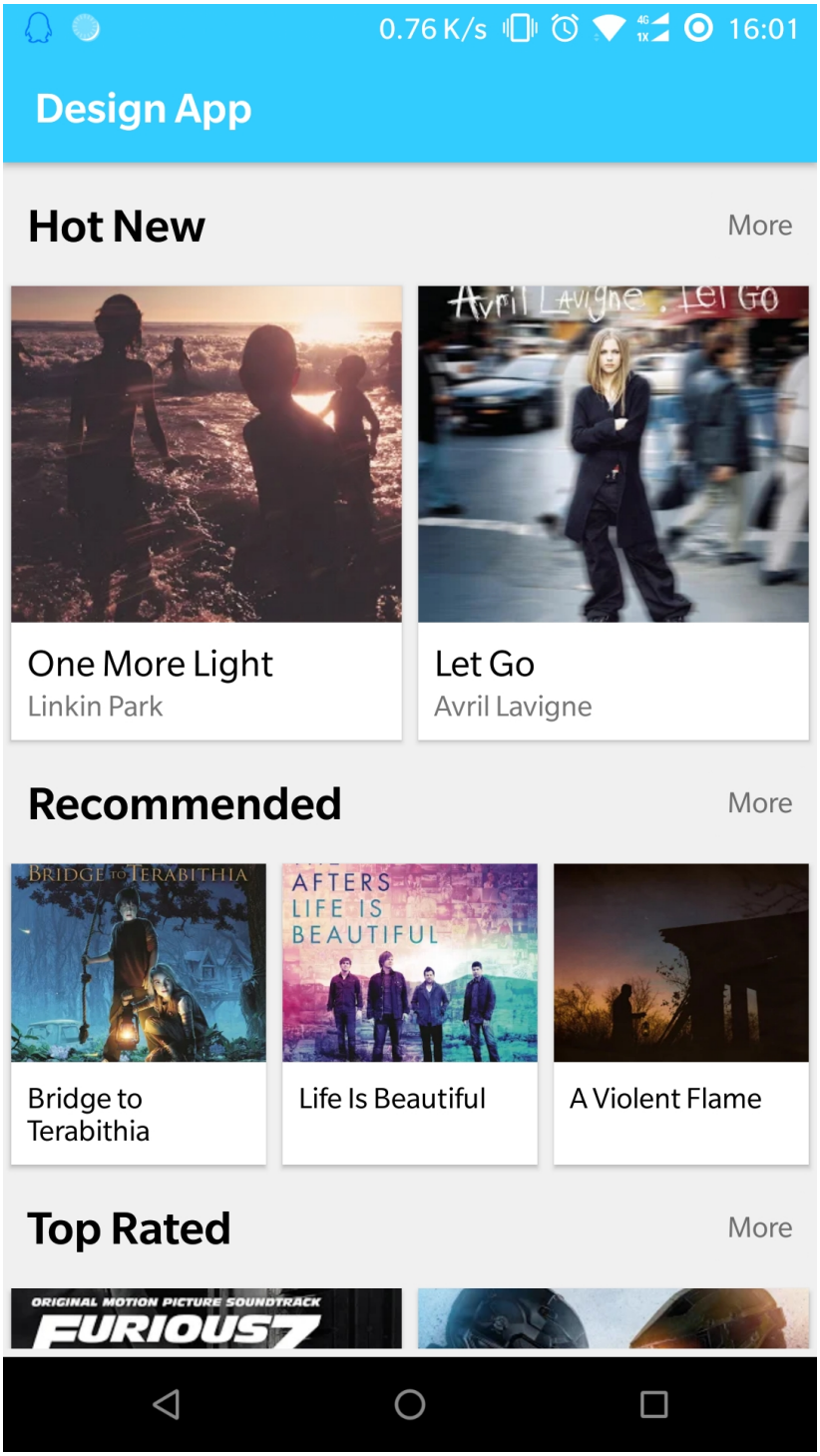
Android RecycleView多种布局实现（工厂模式）

RecycleView是个很常用的控件，很多APP中都可以看到它的身影，同时它也是个很难用的控件，主要就难在多种布局的实现。

在《第一行代码—Android》这本书里边有个RecycleView实现的聊天界面布局，左右两种布局写在了同一个文件中，如果是发送来的消息，就隐藏右侧布局，反之隐藏左侧布局，这种方式对于比较简单的、只有两种Item的界面是可行的，假如我们的Item有多种布局，那么这种方式就显得很笨重。对于多种布局，我们可以使用工厂模式来实现。

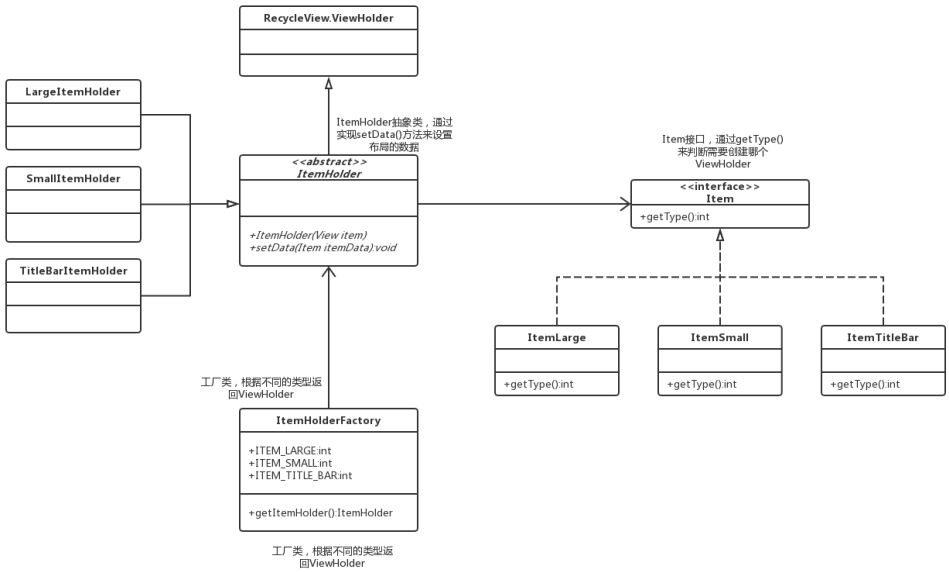
Github:https://github.com/imcloudfloating/DesignApp

1.首先看看效果（GIF一直上传失败，只好传JPG了）：



这里的LayoutManager使用GridLayoutManager，设置为6列，然后在Adapter类中根据不同的类型来设置所占列数，具体见Adapter类的setSpanCount方法。

2.然后是类图：



3.Adapter类：

适配器的代码很短，设置数据和绑定View的代码都写在了ItemHolder的子类里面；

List<Item>储存三种类型的Item数据，如果需要增加新的类型，只要实现Item接口就可以了；

在onBindViewHolder方法中调用ItemHolder的setData()方法来设置数据；

```
public class MultiListAdapter extends RecyclerView.Adapter<ItemHolder> {

    private List<Item> mDataList;

    public MultiListAdapter(List<Item> dataList) {
        mDataList = dataList;
    }

    @NonNull
    @Override
    public ItemHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int type) {
        return ItemHolderFactory.getItemHolder(viewGroup, type);
    }

    @Override
    public void onBindViewHolder(@NonNull ItemHolder viewHolder, int i) {
        //设置 Holder 数据
        viewHolder.setData(mDataList.get(i));
    }

    @Override
    public int getItemViewType(int position) {
        return mDataList.get(position).getType();
    }

    @Override
    public int getItemCount() {
        return mDataList.size();
    }

    public void setSpanCount(GridLayoutManager layoutManager) {
        layoutManager.setSpanSizeLookup(new GridLayoutManager.SpanSizeLookup() {
            @Override
            public int getSpanSize(int i) {
                int type = getItemViewType(i);
                switch (type) {
                    default:
                    case ItemHolderFactory.ITEM_LARGE:
                        return 3;
                    case ItemHolderFactory.ITEM_SMALL:
                        return 2;
```

```
                    case ItemHolderFactory.ITEM_TITLE_BAR:
                        return 6;
                }
            }
        });
    }
}
```

4.ItemHolder抽象类：

setData方法用来设置Item布局的数据

```
public abstract class ItemHolder extends RecyclerView.ViewHolder {
    public ItemHolder(View item) {
        super(item);
    }

    public abstract void setData(Item itemData);
}
```

5.LargeItemHolder类：

另外两个类似

```
public class LargeItemHolder extends ItemHolder {

    private ImageView mItemImage;
    private TextView mTitle;
    private TextView mSubTitle;

    public LargeItemHolder(View item) {
        super(item);
        mItemImage = item.findViewById(R.id.item_image);
        mTitle = item.findViewById(R.id.item_title);
        mSubTitle = item.findViewById(R.id.item_sub_title);
    }

    @Override
    public void setData(Item itemData) {
        ItemLarge item = (ItemLarge) itemData;
        mItemImage.setImageBitmap(item.getImage());
        mTitle.setText(item.getTitle());
        mSubTitle.setText(item.getSubTitle());
    }
}
```

6.Item接口：

```
public interface Item {
    int getType();
}
```

7.ItemLarge类（一个图片、一个标题和一个副标题）：

另外两个类似

```
public class ItemLarge implements Item {

    private Bitmap mImage;
    private String mTitle;
    private String mSubTitle;

    public ItemLarge(Bitmap bitmap, String title, String subTitle) {
        mImage = bitmap;
        mTitle = title;
        mSubTitle = subTitle;
    }
```

```
12
13      public Bitmap getImage() {
14          return mImage;
15      }
16
17      public String getTitle() {
18          return mTitle;
19      }
20
21      public String getSubTitle() {
22          return mSubTitle;
23      }
24
25      @Override
26      public int getType() {
27          return ItemHolderFactory.ITEM_LARGE;
28      }
29  }
```

8.工厂类ItemHolderFactory：

三个常量表示不同的布局类型，通过getItemHolder来创建ViewHolder。

```
1  public class ItemHolderFactory {
2
3      public static final int ITEM_LARGE = 0;
4      public static final int ITEM_SMALL = 1;
5      public static final int ITEM_TITLE_BAR = 2;
6
7      @IntDef({
8              ITEM_LARGE,
9              ITEM_SMALL,
10             ITEM_TITLE_BAR
11     })
12     @interface ItemType {}
13
14     static ItemHolder getItemHolder(ViewGroup parent, @ItemType int type) {
15         switch (type) {
16             default:
17             case ITEM_LARGE:
18                 return new LargeItemHolder(LayoutInflater
19                         .from(parent.getContext()).inflate(R.layout.item_large, parent, false));
20             case ITEM_SMALL:
21                 return new SmallItemHolder(LayoutInflater
22                         .from(parent.getContext()).inflate(R.layout.item_small, parent, false));
23             case ITEM_TITLE_BAR:
24                 return new TitleBarItemHolder(LayoutInflater
25                         .from(parent.getContext()).inflate(R.layout.item_title_bar, parent, false)
26         }
27     }
28 }
```

9.ListActivity类：

```
1  public class ListActivity extends AppCompatActivity {
2
3      List<Item> itemList = new ArrayList<>();
4
5      @Override
6      protected void onCreate(Bundle savedInstanceState) {
7          super.onCreate(savedInstanceState);
8          setContentView(R.layout.activity_list);
9
10         initData();
11
12         GridLayoutManager layoutManager = new GridLayoutManager(this, 6);
13         MultiListAdapter adapter = new MultiListAdapter(itemList);
14         adapter.setSpanCount(layoutManager);
15
```

```
        RecyclerView recyclerView = findViewById(R.id.recycle_view);
        recyclerView.setLayoutManager(layoutManager);
        recyclerView.setAdapter(adapter);
    }

    private void initData() {
        //添加数据
        itemList.add(new ItemTitleBar("Hot New", null));
        itemList.add(new ItemLarge(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_1),
                "One More Light",
                "Linkin Park"));
        itemList.add(new ItemLarge(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_2),
                "Let Go ",
                "Avril Lavigne"));
        itemList.add(new ItemTitleBar("Recommended", null));
        itemList.add(new ItemSmall(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_3),
                "Bridge to Terabithia"));
        itemList.add(new ItemSmall(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_4),
                "Life Is Beautiful"));
        itemList.add(new ItemSmall(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_5),
                "A Violent Flame"));
        itemList.add(new ItemTitleBar("Top Rated", null));
        itemList.add(new ItemLarge(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_6),
                "Furious 7: Original Motion Picture Soundtrack",
                "Various Artists"));
        itemList.add(new ItemLarge(
                BitmapFactory.decodeResource(getResources(), R.drawable.img_7),
                "Halo 5: Guardians (Original Soundtrack)",
                "Kazuma Jinnouchi"));
    }
}
```

10.布局文件（item_large.xml）：

layout_width用match_parent是为了Item在网格中居中，此处match_parent相当于宽度为Item所占的列数。

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="4dp"
    android:background="#ffffff"
    android:elevation="2dp">

    <ImageView
        android:contentDescription="@id/item_title"
        android:id="@+id/item_image"
        android:layout_width="match_parent"
        android:layout_height="170dp"
        android:scaleType="centerCrop"
        tools:src="@drawable/img_7" />

    <TextView
        android:id="@+id/item_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:paddingStart="8dp"
        android:paddingEnd="8dp"
        android:lines="1"
        android:ellipsize="end"
        android:textColor="#000000"
        android:textSize="18sp"
```

```
30          tools:text="Item Title" />
31
32      <TextView
33          android:id="@+id/item_sub_title"
34          android:layout_width="match_parent"
35          android:layout_height="wrap_content"
36          android:layout_marginBottom="8dp"
37          android:paddingStart="8dp"
38          android:paddingEnd="8dp"
39          android:lines="1"
40          android:ellipsize="end"
41          android:textSize="14sp"
42          tools:text="Sub Title" />
43
44  </LinearLayout>
```