官方文档链接：<https://developer.android.com/reference/android/support/v7/widget/RecyclerView.html>

<https://developer.android.com/guide/topics/ui/layout/recyclerview.html>

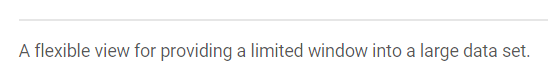
推荐链接：<https://www.kymjs.com/code/2016/07/10/01/>

**我也不知道该怎么介绍这个控件，并且一下子也不可能完全研究透彻，所以只能是用到了就研究一下吧。**

**结合具体的Demo来记录学习内容：https://github.com/DakTop/android-amazing-recycleview**

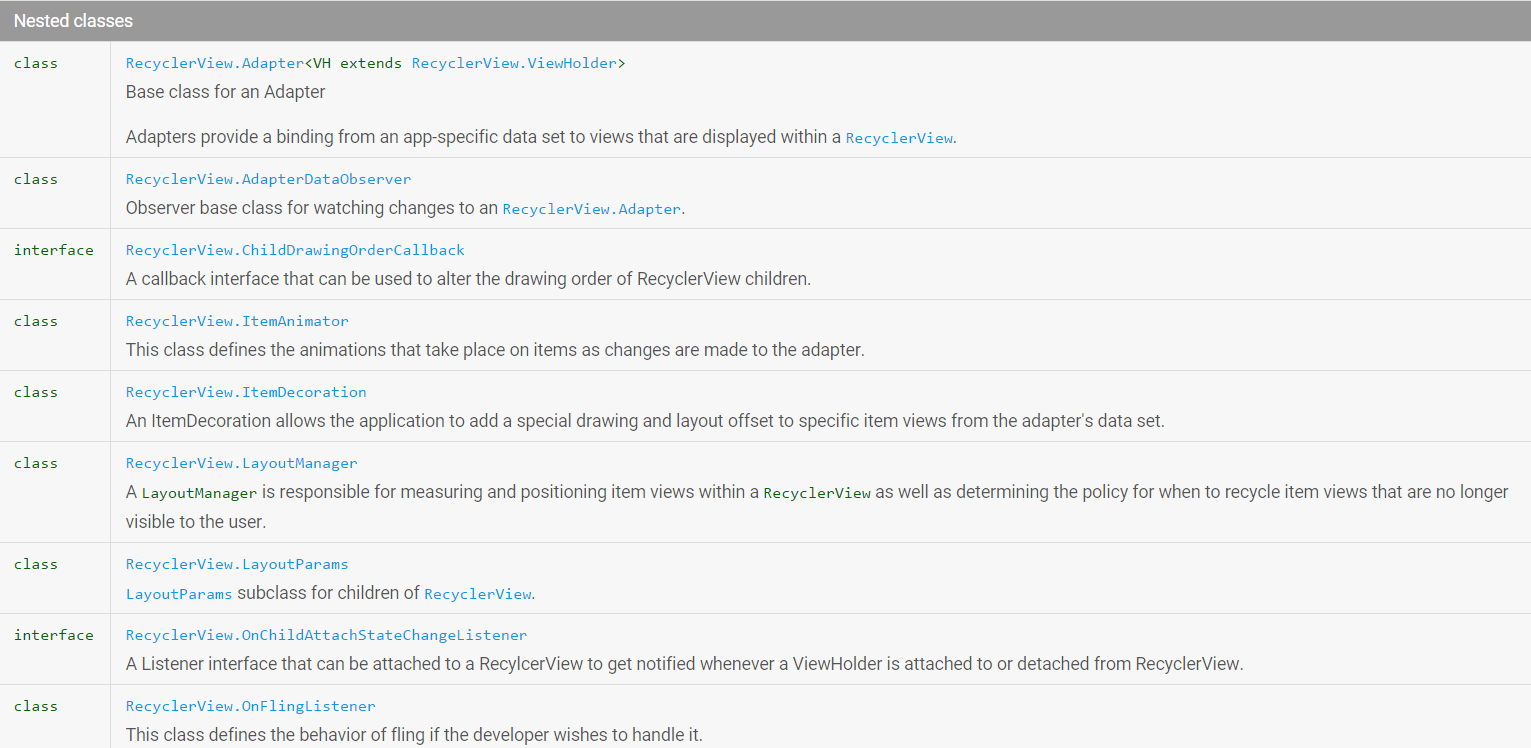
* **官方介绍**

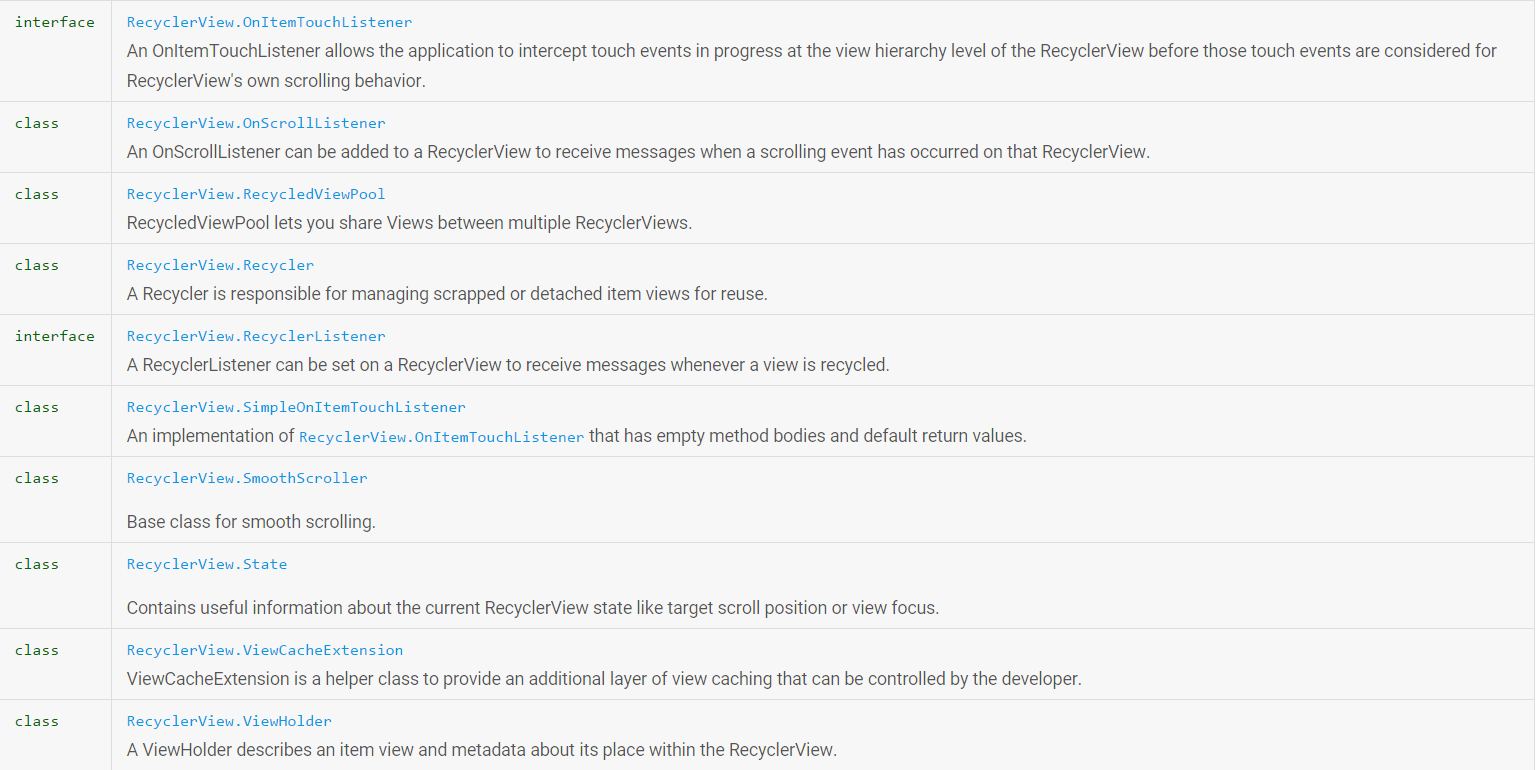
RecyclerView 是 materials design 中的组件之一。

官方定义：

翻译过来就是：为大数据集**提供有限窗口**的**灵活布局**。

* 官方文档中给出涉及到的类：

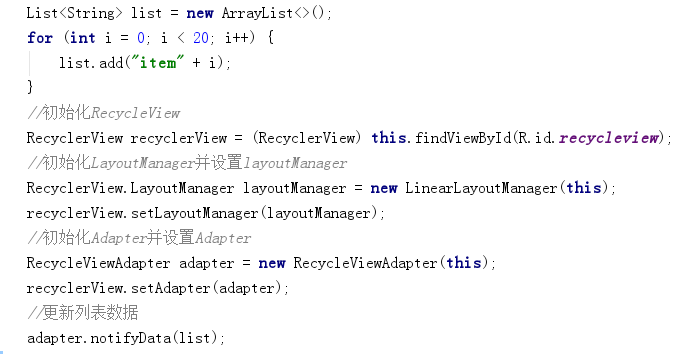




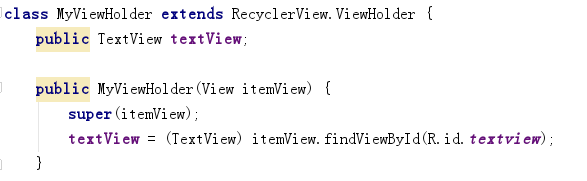
下面就先简单的介绍它的使用，然后一点点剖析上面的类。

* **基础使用**

RecycleView是一个提供有限的窗口循环展示大量数据集的一个View（这句话要牢记）。下面是一个使用RecycleView纵向列表的基础使用：







上面的使用中接触了三个了类：RecycleView、RecycleView.LayoutManager、RecycleView.Adapter、RecycleView.ViewHolder。

RecycleView：继承自ViewGroup，是一个用来装载子视图容器，将ViewHolder中的视图展示。

RecycleView.LayoutManager：负责测量和定位itemview,负责定义屏幕的滚动行为，以及确定何时回收不在可见的itemview。

RecycleView.Adapter：创建ViewHolder，并将数据集绑定到ViewHolder上。

RecycleView.ViewHolder：它管理着RecycleView中的itemView视图和元数据。

它们的作用关系如下如：

RecycleView

Data数据集

Adapter

viewholder

LayoutManager

viewholder.itemview

viewholder.itemview

viewholder

viewholder

viewholder

viewholder.itemview

viewholder

viewholder.itemview

* RecycleView.Adapter

简单说一下Adapter的工作流程:

1. 首先Adapter会调用getItemCount()方法，RecycleView会询问adapter拥有列表包含多少个元素。

2、RecycleView调用onCreateViewHolder()方法创建ViewHolder,以及ViewHolder管理的视图。

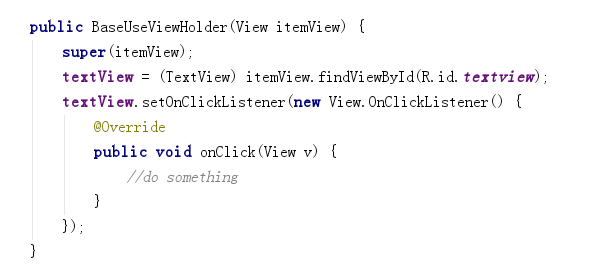
3、每当调用完一次onCreateViewHolder()方法RecycleView会传入ViewHolder及通过LayoutManager来确定其位置，并调用onBindViewHolder()方法绑定相应位置上的数据到ViewHolder。

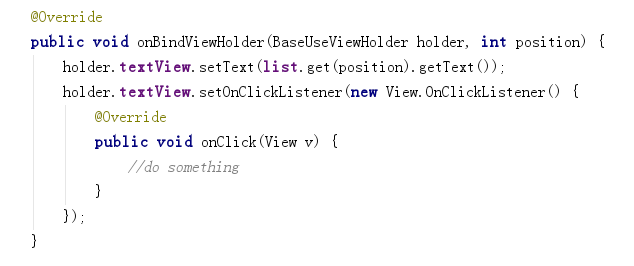
说明：这里并不是数据集合中有多少数据就会创建多少ViewHolder。RecycleView的复用机制就在这里：ViewHolder的创建数量就是你在有限的窗口内所展示itemView的最大数量。比如你的RecycleView设置的大小正好能展示5个itemVeiw，那么RecycleView只会创建5个ViewHolder（也就是说onCreateViewHolder()方法只会执行5次），当列表滚动时，滑出屏幕的ViewHolder会被回收，下面新滑进屏幕的item就是复用刚刚滑出屏幕的ViewHolder并且调用onBindViewHolder()方法重新绑定viewHolder的数据，这样就可以通过回收旧的ViewHolder来节省时间和内存。

* RecycleView的item点击事件

1. **直接设置itemView中的view的点击事件**

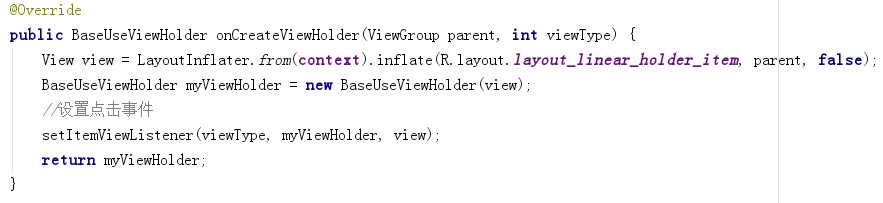
这种方式就是直接在ViewHolder中或者在Adapter中的onBindViewHolder()方法里面设置子View的OnClickListener()，这样无疑耦合性太大，还有就是onClick()处理的内容被放在了ViewHolder中或者Adapter中，这样在使用的时候会导致一些业务逻辑处理起来难度加大：

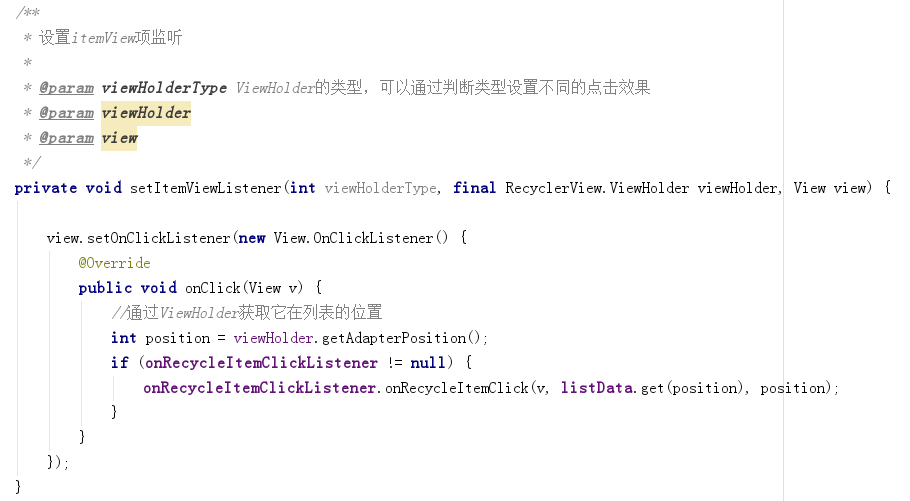




1. **在Adapter中设置ViewHolder管理的布局点击事件**

首先在onCreateViewHolder()方法中设置itemView的点击事件，其中通过RecycleView.ViewHolder中的方法getAdapterPosition()方法获取ViewHolder的位置，即可以获取到相应ViewHolder上绑定的数据：





这样就可以将itemView点击事件的处理内容解耦出去了，在初始化Adapter的地方直接设置它的OnItemClickListener就可以了。

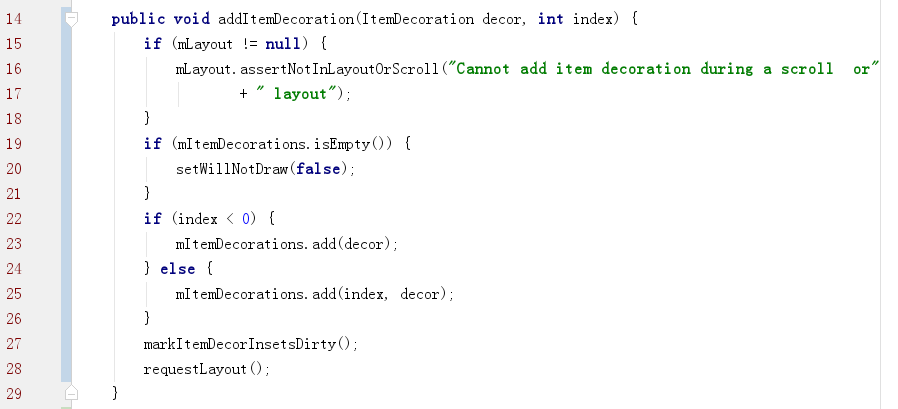
* RecycleView的间隔线RecycleView.ItemDecoration



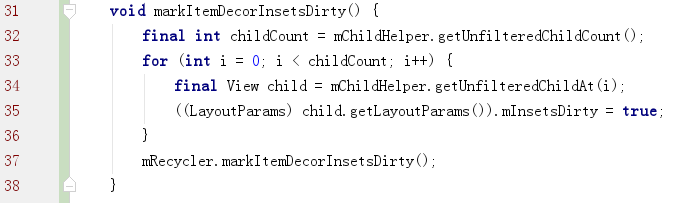
ItemDecoration被官方称为装饰，就是说可以通过ItemDecoration这个类你可以在RecycleView子View的上下左右装饰任何内容。

想要设置装饰，需要重写ItemDecoration类，然后重写它里面的三个方法：OnDraw，onDrawOver，getItemOffsets三个方法，这三个方法的用法都已在上图中标明。

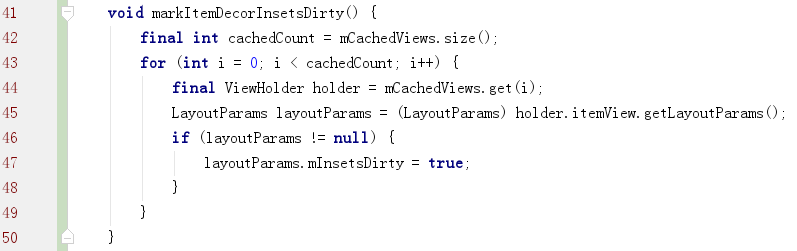
RecycleView通过addItemDecoration添加装饰，从这个方法开始深入解析ItemDecoration的添加过程：



这个方法很简单，只看重点，参数index代表这个装饰添加的位置，通过L23或者L25将装饰添加到mItemDecorations中，mItemDecorations是个ArrayList<ItemDecoration>集合对象，这就意味着我们可以通过RecycleView的addItemDecoration方法按照添加顺序添加多个ItemDecoration装饰，将装饰添加进集合后，L27调用markItemDecorInsetsDirty：

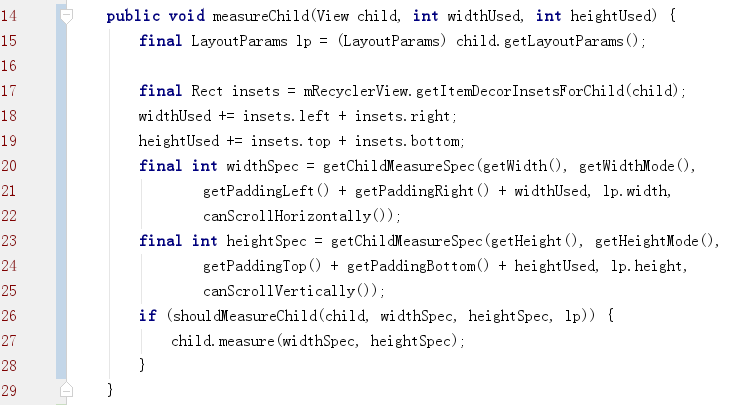


这个方法很简单，首先通过mChildHelp遍历RecycleView中所有的子View，然后在L35将子View的LayoutParams中的mInsetsDirty设置为true，这里有两点:第一,子View设置的LayoutParams是继承自MarginLayoutParams的子类，是RecycleView的内部类，专门为ItemDecoration而设计。第二，mInsetsDirty设置为true证明RecycleView的装饰有改动，当RecycleView测量子View的时候通过这个字段来决定是否使用LyoutParams缓存中的装饰设置的距离值，后面会看到。最后L37调用mRecycler.markItemDecorInsetsDirty方法，Recycler 是 RecyclerView 的一个内部类，它管理着RcycleView的复用逻辑：

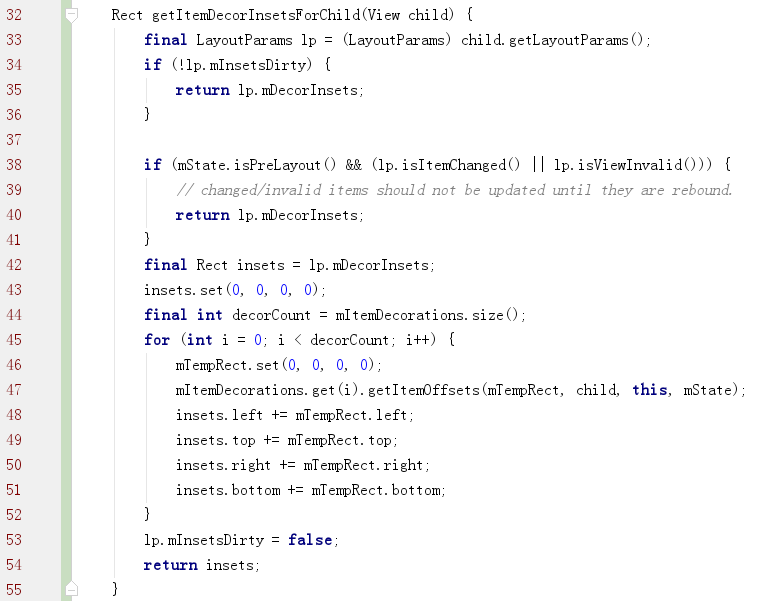


mRecycler.markItemDecorInsetsDirty方法也很简单，它遍历缓存中所有的子View最后在L47行，将缓存中的mInsetsDirty设置为true。

当上面这个方法执行完了就会执行addItemDecoration方法中最后一行的requestLayout()方法重新测量，布局，绘制RecycleView。在重新测量，布局，绘制过程中会调用到ItemDecoration类中OnDraw，onDrawOver，getItemOffsets三个方法。首先在测量过程中会调用measureChild方法：



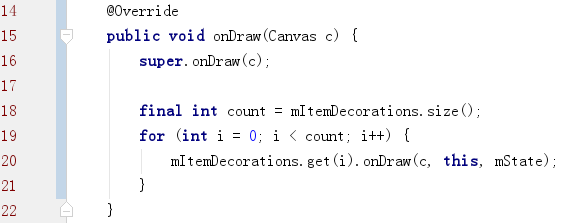
L17调用RecycleView的getItemDecorInsetsForChild方法并将子View传递进去：



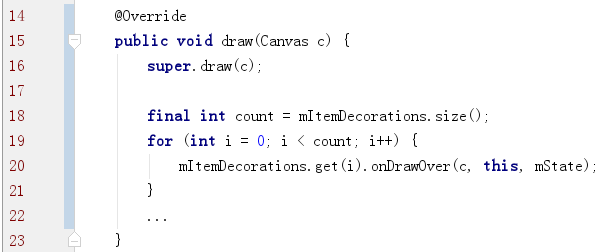
看L33行获取到子View的LayoutParams并在L34判断它的mInsetsDirty值，在上面分析RecycleView的addItemDecoration方法时知道，当我们成功添加一个ItemDecoration的时候，RecycleView会将它内部包括缓存中所有的子View的mInsetsDirty值都设置为true，所以mInsetsDirty为true则需要重新计算RecycleView中所有装饰所占的空间，L42获取子View中Rect对象并重置，接下来遍历mItemDecorations集合中的全部的装饰，并且调用ItemDecoration的getItemOffsets方法获取它们的left,top,right,bottom并累加到子View的LayoutParams的Rect对象中并返回。最后L53设置mInsetsDirty为false，这时候如果RecycleView再次测量，而它的ItemDecoration装饰没有变化，则不会再重新计算ItemDecoration的值了。

接下来继续看measureChild那张代码图（也就是上上个图），通过L17调用getItemDecorInsetsForChild方法返回计算的Rect后，接下来就通过这些值计算RecycleView的子View的大小了。这样RecycleView的一个子View就测量完成了，每当测量一个子View就会计算所有的ItemDecoration的left,top,right,bottom值。

测量完了，最后绘制的过程中调用onDraw方法：



L20会调用ItemDecoration的onDraw方法，在RecycleView的item绘制之前调用。然后在RecycleView的draw方法中调用ItemDecoration的onDrawOver方法，onDrawOver是在在RecycleView的item绘制之后调用：



。

* **疑难杂症**

解决在开发中遇到的一些问题。

* ScrollView嵌套RecycleView时

使用ScrollView嵌套RecycleView时会遇到RecycleView显示不全的问题，这时候需要使用到RecycleLayout的descendantFocusability属性，具体做法就是在RecycleView外面在套一层RecycleView并设置属性android:descendantFocusability="blocksDescendants"。这里descendantFocusability属性一共有三个可选项：

beforeDescendants：viewGroup会优先子类控件而获取焦点

afterDescendants：viewGroup只有当子类控件不需要获取焦点的时候才去获取焦点

blocksDescendants：viewGroup会覆盖子类控件而直接获取焦

另外，RecycleView嵌套RecycleView不用做处理就可以完美嵌套，也不用担心滑动事件和点击事件的问题。