Android 滚动篇

工程师 李洪江

一 NestedScrollView 嵌套 RecycleView 发生的小问题

1、解决方法：嵌套滑动不激活。

recycleView.setNestedScrollingEnable(false); 这样做有个弊端，RecycleView 的 item 会一次性加载完，不管是否显示，如果 item 比较多的话不建议这样使用布局，如果再有下拉刷新，这种布局设计是非常糟糕的。

建议：只使用 RecycleView，通过设置 item 类型来显示要展示的布局

比如 itemType = 0 填充 Banner 布局

itemType = 1 填充菜单布局

....等等

2、当 NestedScrollView 嵌套 RecycleView 布局由 Fragment 管理，Fragment 切换时会自动滑动到 ReycleView 的顶部。

解决方法：在 NestedScrollView 唯一子布局中加入 android:descendantFocusability=“blocksDescendants”

android:descendantFocusability 有三个属性

优先于子控件获取焦点“beforeDescendants"

当子控件不需要焦点时，获取焦点”afterDescendants“

覆盖所有子控件获取焦点，”blocksDescendants“。

二 对于滚动嵌套处理，常见内部测量布局，外部滚动事件拦截。

参考：http://blog.csdn.net/ganyao939543405/article/details/52204992

三 Scrollview (NestedScrollView) 嵌套 RecyclerView 的时候 RecyclerView 抢焦点

1 Scrollview (NestedScrollView) 嵌套 RecyclerView 的时候 RecyclerView 抢焦点，跳转到这个 Activity 页面的时候，Scrollview 自动滑动到 RecyclerView 的地方而不是本页面的最上方的 View， 这时候是因为 RecyclerView 抢了焦点 ，自动滑动，只需要在 xml 页面在最顶层的 View 加入

android:focusableInTouchMode="true"
android:focusable="true"

让最上端的 View 获取焦点 即可。

2 内部数据更新，这导致了 ScrollView 内部重新走了 onLayout / onMeaure 流程 在这个流程中 ScrollView 会将自身滚动到 获得 focus 的 child 位置。

遇到过一个类似的问题,不过当时 `ScrollView` 里嵌套的是 `GridView` ,分享下解决方案及过程,只想要解决方案的话看分隔线上面的就好了

- **grid.setFocusable(false)**
- **update数据后,手动scrollto(0,0)**
- **重写ScrollView中的computeScrollDeltaToGetChildRectOnScreen,让该方法返回0**

`ScrollView` 中嵌套 `GridView` 导致 `ScrollView` 默认不停留在顶部的解决方案和分析

发生情况大概是我在ScrollView顶部放了一个ViewPager用来做广告Banner,底部放了个GridVie, 来实现一个类似9宫格效果的展示.

然后出现的状况是,当我获取完数据并调用notifyDataSetChanged();后 ScrollView自动滚到了最底部,也就是GridView所在的位置.

研究了一下,获取了一些解决方案

-- 让界面顶部的某一个View获取focus
-- grid.setFocusable(false);
-- 手动scrollto(0,0)
-- 重写ScrollView中的computeScrollDeltaToGetChildRectOnScreen,让该方法返回0

目前简单的用setFocusable(false)解决了该问题

试着分析一下这个问题产生的原因. 从解决方案反推,可以发现这个问题和 `focus` 有关系

一个猜测是 notifyDataSetChanged()之后,grid由于加载了数据的关系高度产生了变化

这导致了ScrollView内部重新走了 onLayout / onMeaure 流程 在这个流程中 ScrollView会将自身滚动到 获得 focus 的 child 位置

上面关于focus的解决方案即是从这个角度去解决问题

跟踪一下调用链

```
protected void onLayout(boolean changed, int l, int t, int r, int b) {
 super.onLayout(changed, l, t, r, b);
 mIsLayoutDirty = false;
 // Give a child focus if it needs it
 if (mChildToScrollTo != null && isViewDescendantOf(mChildToScrollTo, this)) {
 scrollToChild(mChildToScrollTo);
 }
 ...
}
```

可以看到 onLayout 的时候确实会将ScrollView滚动到focus child位置

```
private void scrollToChild(View child) {
 child.getDrawingRect(mTempRect);

 /* Offset from child's local coordinates to ScrollView coordinates */
 offsetDescendantRectToMyCoords(child, mTempRect);

 int scrollDelta = computeScrollDeltaToGetChildRectOnScreen(mTempRect);

 if (scrollDelta != 0) {
 scrollBy(0, scrollDelta);
 }
}
```

从代码逻辑上来看 避免 GridView获取focus可以解决该问题.

而重写ScrollView中的computeScrollDeltaToGetChildRectOnScreen则是从另一个角度解决该问题

而scrollToChild这个方法会根据computeScrollDeltaToGetChildRectOnScreen的返回值来计算滚动的位置

重载computeScrollDeltaToGetChildRectOnScreen让其返回0 会导致ScrollView内布局产生变化时,不能正确滚动到focus child位置,当然,这也就是我们想要的效果,布局变化时ScrollView不需要自己去滚动.

至于computeScrollDeltaToGetChildRectOnScreen代码太长就不贴了

大致代码是 根据当前 scrollY和focus child 的 rect.bottom 去计算要滚到哪

逻辑理顺以后觉得这个问题也没什么奇怪的.

四 去掉 GridView 的滚动机制，达到不复用。

```
public class NoScrollGridView extends GridView {

    public NoScrollGridView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public NoScrollGridView(Context context) {
        super(context);
    }

    public NoScrollGridView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
    }

    public void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        //核心在此
        int expandSpec = MeasureSpec.makeMeasureSpec(Integer.MAX_VALUE >> 2,
                MeasureSpec.AT_MOST);
        super.onMeasure(widthMeasureSpec, expandSpec + 50);
    }
}
```

去掉内容改变，滚动到焦点位置

```
public class NoScrollView extends ScrollView {

    public NoScrollView(Context context) {
        super(context);
    }

    public NoScrollView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public NoScrollView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
    }

    @Override
    protected int computeScrollDeltaToGetChildRectOnScreen(Rect rect) {
        return 0;
    }

}
```