你必须了解的 RecyclerView 的五大开源项目 - 解决上拉加载、下拉刷新和添加 Header、Footer 等问题

gold.xitu.io

前段时间做项目由于采用的MD设计，所以必须要使用RecyclerView全面代替ListView。但是开发中遇到了需要实现RecyclerView上拉加载、下拉刷新和添加Header以及Footer等需求问题，现将问题解决中用到的五大开源项目总结下来，方便他人。

首先介绍下**RecyclerView，RecyclerView**相比**ListView**增加了很多新特性：

• Adapter中的ViewHolder模式 - 对于ListView来说，通过创建ViewHolder来提升性能并不是必须的。因为ListView并没有严格的ViewHolder设计模式。但是在使用RecyclerView的时候，Adapter必须实现至少一个ViewHolder，必须遵循ViewHolder设计模式。
• 定制Item条目 - ListView只能实现垂直线性排列的列表视图，与之不同的是，RecyclerView可以通过设置RecyclerView.LayoutManager来定制不同风格的视图，比如水平滚动列表或者不规则的瀑布流列表。
• Item动画 - 在ListView中没有提供任何方法或者接口，方便开发者实现Item的增删动画。相反地，可以通过设置RecyclerView的RecyclerView.ItemAnimator来为条目增加动画效果。
• 设置数据源 - 在LisView中针对不同数据封装了各种类型的Adapter，比如用来处理数组的ArrayAdapter和用来展示Database结果的CursorAdapter。相反地，在RecyclerView中必须自定义实现RecyclerView.Adapter并为其提供数据集合。
• 设置条目分割线 - 在ListView中可以通过设置android:divider属性来为两个Item间设置分割线。如果想为RecyclerView添加此效果，则必须使用RecyclerView.ItemDecoration，这种实现方式不仅更灵活，而且样式也更加丰富。
• 设置点击事件 - 在ListView中存在AdapterView.OnItemClickListener接口，用来绑定条目的点击事件。但是，很遗憾的是在RecyclerView中，并没有提供这样的接口，不过，提供了另外一个接口RcyclerView.OnItemTouchListener，用来响应条目的触摸事件。
但是……，RecyclerView不像ListView那样拥有Header和Footer，因此开发中需要我们自己去实现Header和Foote，另外开发中小伙伴们经常使用的PullToRefresh库暂时又不支持RecyclerView。和身边的很多小伙们一样，我也陷入了困境，为了不拖累项目进度，我决定亲自解（shi）决（yong）难（kai）题（yuan），做一个伸手党。

现在将我发现的GitHub上优秀的Header、Footer、上拉加载和下拉刷新解决方案汇总如下：

（一） SwipeToLoadLayout-推荐使用

GitHub地址：https://github.com/Aspsine/SwipeToLoadLayout

> SwipeToLoadLayout支持YouTube、Google、京东等多家APP基于RecyclerView的上拉加载和下拉刷新样式，，好用的不要不要的。废话不多说，直接上图：

• ListView & GridView

• RecyclerView(With all kinds of layoutManagers)

• WebView & ScrollView & Other Views

• Google SwipeRefreshLayout style

• 京东style

• Yalantis Phoenix 样式

AndroidStudio配置方法

第一步：在你的build.gradle添加JitPack库在
repositories {
maven { url “https://jitpack.io” }
}

第二部：添加依赖库
dependencies {
compile ‘com.github.Aspsine:SwipeToLoadLayout:v1.0.2’
}

（二） UltimateRecyclerView-大名鼎鼎

GitHub地址：https://github.com/cymcsg/UltimateRecyclerView

> UltimateRecyclerView是解决RecyclerView下拉刷新，加载更多，增加头部，显示或隐藏工具栏等许多问题的知名开源框架。

包含特性如下：
• Swipe to refresh(using android.support.v4.widget.SwipeRefreshLayout)
• Many kinds of animations
• Swipe to dismiss
• Parallax or normal head view
• Drag and drop items
• Loading more when reach the last item(infinite scrolling)
• Custom views in loading more
• Showing or hiding toolbar and floating button when scrolling
• Scrollbars
• Colorful styles of swipe to refresh
• Sticky header like instagram
• Support different layout in adapter
• Loading adapter with animation

使用效果如下：

AndroidStudio配置方法

第一步：在你的build.gradle添加库

```
repositories {
    jcenter()
    maven { url "http://dl.bintray.com/jjhesk/maven" }
    }
```

第二步：添加依赖库

```
dependencies{
    compile 'com.hkm.slidingmenulib:libmenu:0.4.9'
}
```

第三步：布局文件中使用方法

```
<com.marshalchen.ultimaterecyclerview.UltimateRecyclerView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/ultimate_recycler_view"
/>
```

（三） IRecyclerView-效果最炫

GitHub地址：https://github.com/Aspsine/IRecyclerView

```
IRecyclerView支持RecyclerView下拉刷新，上拉加载，定制Header和Footer。
```

包含特性如下：
• pull-to-refresh
• pull-to-loadmore
• customize refresh header
• customize loadmore footer
• add multiple header view
• add multiple footer view

使用效果如下：
刷新效果

AndroidStudio配置方法
第一步：在你的build.gradle添加库

```
repositories:
allprojects {
    repositories {
        ...
        maven { url "https://jitpack.io" }
    }
}
```

第二步：添加依赖库

```
dependencies {
    compile 'com.github.Aspsine:IRecyclerView:0.0.2'
}
```

第三步：布局文件中使用方法

```
<com.aspsine.irecyclerview.IRecyclerView
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/iRecyclerView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:loadMoreEnabled="true"

app:loadMoreFooterLayout="@layout/layout_irecyclerview_load_more_footer
"
    app:refreshEnabled="true"

app:refreshHeaderLayout="@layout/layout_irecyclerview_refresh_header"/>
```

第四步：Activity/Fragment中使用

```
IRecyclerView iRecyclerView = (IRecyclerView)
findViewById(R.id.iRecyclerView);

iRecyclerView.setLayoutManager(new LinearLayoutManager(this));

// an custom footer view, you can customize it yourself.
LoadMoreFooterView loadMoreFooterView = (LoadMoreFooterView)
iRecyclerView.getLoadMoreFooterView();

// you can also add header and footer like this
// note: header and refresh header are different, footer and load more
footer are different too.
iRecyclerView.addHeaderView(headerView);
iRecyclerView.addFooterView(footerView);

// adapter
ImageAdapter mAdapter = new ImageAdapter();
// note: here use setIAdapter(...) method not setAdapter(...)
iRecyclerView.setIAdapter(mAdapter);

iRecyclerView.setOnRefreshListener(new OnRefreshListener() {
    @Override
    public void onRefresh() {

    }
});
iRecyclerView.setOnLoadMoreListener(new OnLoadMoreListener() {
```

```
        @Override
        public void onLoadMore(View loadMoreView) {

        }
    });

    // set auto refreshing
    iRecyclerView.post(new Runnable() {
        @Override
       public void run() {
           iRecyclerView.setRefreshing(true);
       }
    });

    // stop refreshing
    iRecyclerView.setRefreshing(false);
```

（四）PullLoadMoreRecyclerView-属性最全

GitHub地址：https://github.com/WuXiaolong/PullLoadMoreRecyclerView

```
PullLoadMoreRecyclerView实现了RecyclerView下拉刷新和上拉加载更多以及RecyclerView线性、网格、瀑布流效果。
```

效果图如下：

使用方法
build.gradle文件

```
dependencies {
  compile 'com.wuxiaolong.pullloadmorerecyclerview:library:1.0.4'
}
```

xml引用

```
<com.wuxiaolong.pullloadmorerecyclerview.PullLoadMoreRecyclerView
        android:id="@+id/pullLoadMoreRecyclerView"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:layout_margin="10dp" />
```

设置线性布局

```
 mPullLoadMoreRecyclerView = (PullLoadMoreRecyclerView)
view.findViewById(R.id.pullLoadMoreRecyclerView)
 mPullLoadMoreRecyclerView.setLinearLayout()
```

设置网格布局

```
 mPullLoadMoreRecyclerView.setGridLayout(2);//参数为列数
```

设置交错网格布局，即瀑布流效果

```
 mPullLoadMoreRecyclerView.setStaggeredGridLayout(2);//参数为列数
```

绑定适配器

```
mRecyclerViewAdapter = new RecyclerViewAdapter();
mPullLoadMoreRecyclerView.setAdapter(mRecyclerViewAdapter);

public class RecyclerViewAdapter extends
RecyclerView.Adapter<recyclerviewadapter.viewholder>
{</recyclerviewadapter.viewholder>

    public RecyclerViewAdapter() {

    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int
viewType) {
        View view =
LayoutInflater.from(parent.getContext()).inflate(R.layout.recycler_view
_item, parent, false);
        return new ViewHolder(view);
    }
```

```
    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {

    }

    @Override
    public int getItemCount() {
        return 0;
    }

    public class ViewHolder extends RecyclerView.ViewHolder {

        public ViewHolder(View itemView) {
            super(itemView);

        }
    }
}
```

调用下拉刷新和加载更多

```
mPullLoadMoreRecyclerView.setOnPullLoadMoreListener(new
PullLoadMoreRecyclerView.PullLoadMoreListener() {
            @Override
            public void onRefresh() {

            }

            @Override
            public void onLoadMore() {

            }
        });
```

刷新结束

```
mPullLoadMoreRecyclerView.setPullLoadMoreCompleted()
```

不需要下拉刷新

```
mPullLoadMoreRecyclerView.setPullRefreshEnable(false);
```

不需要上拉刷新

```
mPullLoadMoreRecyclerView.setPushRefreshEnable(false);
```

设置上拉刷新文字

```
mPullLoadMoreRecyclerView.setFooterViewText("loading")
```

设置下拉刷新颜色

```
mPullLoadMoreRecyclerView.setColorSchemeResources(android.R.color.holo_
red_dark,android.R.color.holo_blue_dark)
```

快速Top

```
 mPullLoadMoreRecyclerView.scrollToTop()
```

（五）HeaderAndFooterRecyclerView-封装完善

GitHub地址：https://github.com/cundong/HeaderAndFooterRecyclerView

```
HeaderAndFooterRecyclerView是支持addHeaderView、 addFooterView、分页加载的
RecyclerView解决方案。它可以对 RecyclerView 控件进行拓展（通过
RecyclerView.Adapter实现），给RecyclerView增加HeaderView、FooterView，并且
不需要对你的具体业务逻辑Adapter做任何修改。同时，通过修改 FooterView State，可以
动态 FooterView 赋予不同状态（加载中、加载失败、滑到最底等），可以实现
RecyclerView 分页加载数据时的 Loading/TheEnd/NetWorkError 效果。
```

• 添加HeaderView、FooterView
mHeaderAndFooterRecyclerViewAdapter = new
HeaderAndFooterRecyclerViewAdapter(mDataAdapter);
mRecyclerView.setAdapter(mHeaderAndFooterRecyclerViewAdapter);

```
    mRecyclerView.setLayoutManager(new LinearLayoutManager(this));

    //add a HeaderView
    RecyclerViewUtils.setHeaderView(mRecyclerView, new
SampleHeader(this));

    //add a FooterView
    RecyclerViewUtils.setFooterView(mRecyclerView, new
SampleFooter(this));
```

• LinearLayout/GridLayout/StaggeredGridLayout布局的RecyclerView分页加载
mRecyclerView.addOnScrollListener(mOnScrollListener);
private EndlessRecyclerOnScrollListener mOnScrollListener = new
EndlessRecyclerOnScrollListener() {

```
    @Override
    public void onLoadNextPage(View view) {
        super.onLoadNextPage(view);

        LoadingFooter.State state =
RecyclerViewStateUtils.getFooterViewState(mRecyclerView);
        if(state == LoadingFooter.State.Loading) {
            Log.d("@Cundong", "the state is Loading, just wait..");
            return;
        }

        mCurrentCounter = mDataList.size();

        if (mCurrentCounter < TOTAL_COUNTER) {
            // loading more

RecyclerViewStateUtils.setFooterViewState(EndlessLinearLayoutActivity.t
his, mRecyclerView, REQUEST_COUNT, LoadingFooter.State.Loading, null);
            requestData();
        } else {
            //the end

RecyclerViewStateUtils.setFooterViewState(EndlessLinearLayoutActivity.t
his, mRecyclerView, REQUEST_COUNT, LoadingFooter.State.TheEnd, null);
        }
```

```
        }
    };
```

注意事项

> 如果已经使用 RecyclerViewUtils.setHeaderView(mRecyclerView, view); 为RecyclerView添加了HeaderView，那么再调用ViewHolder类的getAdapterPosition()、getLayoutPosition()时返回的值就会因为增加了Header而受影响（返回的position等于真实的position+headerCounter）。
> 因此，这种情况下请使用 RecyclerViewUtils.getAdapterPosition(mRecyclerView, ViewHolder.this)、RecyclerViewUtils.getLayoutPosition(mRecyclerView, ViewHolder.this) 两个方法来替代。
> 使用效果：

- 添加HeaderView、FooterView
- 支持分页加载的LinearLayout布局RecyclerView
- 支持分页加载的GridLayout布局RecyclerView
- 支持分页加载的StaggeredGridLayout布局RecyclerView
- 分页加载失败时的GridLayout布局RecyclerView

以上就是GitHub中比较好的RecyclerView开源框架，希望能对小伙伴们的开发带来帮助，更感谢这些作者们提供了这么好的东西！

安卓开发高级技术交流QQ群：108721298