**Android 仿知乎广告控件,广告图随滑动控件滑动**

2017年12月20日 15:47:34 [郭辉先森](https://me.csdn.net/sinat_26708935) 阅读数：681

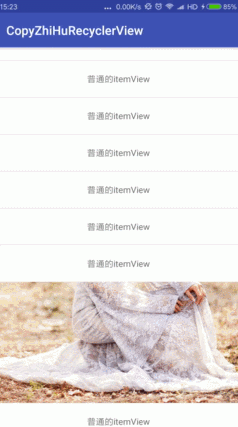
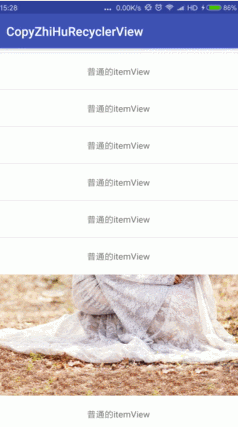
 版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/sinat\_26708935/article/details/78853617

**仿知乎广告模块,效果:RecyclerView其中的一个item是广告图片**

**- 知乎的效果图如下:**

* 从下到上
* 
* 从上到下
* 

**- 仿的效果图:**

* 两种情况,一种是广告图片比滑动控件长,另外一种是广告图片比滑动控件短,效果如下:
* 广告图片比滑动控件短:   
  
* 广告图片比滑动控件长:   
  

**实现思路**

1.通过给RecyclerView设置addOnScrollListener监听监听广告框是否出现在视野中   
2.通过这个方法获取读取对应区域的bitmap对象,其中inputStream是图片的数据流   
BitmapRegionDecoder mDecoder = BitmapRegionDecoder.newInstance(inputStream, true);   
3.通过方法1判定需要读取的区域，用方法2中的mDecoder对象读取对应区域   
mDecoder.decodeRegion(mRect, null);   
其中mRect用来控制读取范围   
private final Rect mRect = new Rect();

**- 代码解析**

* 1.通过给RecyclerView设置了addOnScrollListener()监控滑动

private class OnScrollLisrener extends RecyclerView.OnScrollListener {

@Override

public void onScrolled(RecyclerView recyclerView, int dx, int dy) {

super.onScrolled(recyclerView, dx, dy);

*//获取广告的itemView*

View ggView = linearLayoutManager.findViewByPosition(ggPosition);

if (ggView == null) {

return;

}

if (ggImageView == null) {

return;

}

*//获取滑动控件的高*

parentHeight = mRecyclerView.getHeight();

*//图片距离滑动控件的上下距离*

int topOrBottomPadding;

int top = ggView.getTop();

int left = 0;

int right = imageWidth;

int bottom = ggView.getBottom();

*//如果图片比滑动控件短*

if (parentHeight > imageHeight) {

*//计算图片距离顶部的距离和图片距离底部的距离*

topOrBottomPadding = (parentHeight - imageHeight) / 2;

*//获取item的高*

int itemHeight = ggView.getHeight();

if (top >= parentHeight - itemHeight - topOrBottomPadding) {

*//如果超出底部,就一直显示图片的底部*

bottom = imageHeight;

top = bottom - itemHeight;

} else if (top <= topOrBottomPadding) {

*//如果超出顶部,就一直显示图片的顶部*

top = 0;

bottom = top + itemHeight;

} else {

*//处于图片中的时候,自由滑动*

top -= topOrBottomPadding;

bottom = top + itemHeight;

}

}

mRect.set(left, top, right, bottom);

*//异步（异步时会卡...貌似是因为线程延迟的问题）*

*// executorService.execute(bitmapRunnable);*

*//同步→*

ggBitmap = mDecoder.decodeRegion(mRect, null);

if (ggBitmap != null) {

ggImageView.setImageBitmap(ggBitmap);

}

*//←同步*

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 2.activity中setGGViewPosition()方法是为了让adapter把广告item的position，最好还有广告的ImageView抛出(省的在onScroll里重复去寻找)

*//用来从adapter里设置广告item的位置和ImageView（扩展：从这里传广告的图片地址,然后去加载）*

public void setGGViewPosition(int ggPosition, ImageView imageView) {

this.ggPosition = ggPosition;

this.ggImageView = imageView;

}

* 1
* 2
* 3
* 4
* 5
* 3.adapter中onBindViewHolder()方法中把位置和ImageView设置给Activity,这里可以在方法中再加一个网络图片的链接,在activity中加载图片,广告图片就可以随时变动了

@Override

public void onBindViewHolder(ViewHolder holder, int position) {

int itemViewType = getItemViewType(position);

switch (itemViewType) {

case ItemBean.ITEM\_GG:

GgViewHolder ggHolder = (GgViewHolder) holder;

mainActivity.setGGViewPosition(position, ggHolder.ggView);

break;

case ItemBean.ITEM\_PT:

break;

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14

**- activity代码,里面有详细注释**

里面有一些异步任务代码,这里其实现在不能加载过大的图片,不然可能会卡顿,如果有人优化好了,贴评论里或者给我私信，我会共享给大家

public class MainActivity extends AppCompatActivity {

*//adapter中广告控件里的ImageView*

private ImageView ggImageView;

*//根据位置读取的广告图片*

private Bitmap ggBitmap;

*//广告item所在的位置*

private int ggPosition = -1;

*//滑动控件*

private RecyclerView mRecyclerView;

*//adapter*

private CopyZhiHuAdapter adapter;

*//线性布局*

private LinearLayoutManager linearLayoutManager;

*//用来控制读取范围*

private final Rect mRect = new Rect();

*//读取区域bitmap的类*

private BitmapRegionDecoder mDecoder;

*//需要显示的广告图片的高*

private int imageHeight;

*//需要显示的广告的宽*

private int imageWidth;

*//滑动控件的高*

private int parentHeight;

*//忽略这些，我想异步的处理数据,结果有明显的卡顿*

private ExecutorService executorService;

*//忽略这些，我想异步的处理数据,结果有明显的卡顿*

private Handler mHandler = new Handler() {

@Override

public void handleMessage(Message msg) {

super.handleMessage(msg);

if (ggBitmap != null) {

ggImageView.setImageBitmap(ggBitmap);

}

}

};

*//忽略这些，我想异步的处理数据,结果有明显的卡顿*

private Runnable bitmapRunnable = new Runnable() {

@Override

public void run() {

ggBitmap = mDecoder.decodeRegion(mRect, null);

mHandler.sendEmptyMessage(1);

}

};

*//用来从adapter里设置广告item的位置和ImageView（扩展：从这里传广告的图片地址,然后去加载）*

public void setGGViewPosition(int ggPosition, ImageView imageView) {

this.ggPosition = ggPosition;

this.ggImageView = imageView;

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

adapter = new CopyZhiHuAdapter(this);

mRecyclerView = (RecyclerView) findViewById(R.id.start);

linearLayoutManager = new LinearLayoutManager(this);

mRecyclerView.setLayoutManager(linearLayoutManager);

mRecyclerView.setAdapter(adapter);

mRecyclerView.addOnScrollListener(new OnScrollLisrener());

*//用线程池更新图片*

executorService = Executors.newSingleThreadExecutor();

addJia();

addImageResources();

}

*//假装这里是异步加载的网络图片*

private void addImageResources() {

try {

*//gg\_image2，gg\_image3，gg\_image4,三张图片的高度不一样，顺序为：从长到短*

@SuppressWarnings("ResourceType")

InputStream inputStream = getResources().openRawResource(R.mipmap.gg\_image4);

mDecoder = BitmapRegionDecoder.newInstance(inputStream, true);

imageHeight = mDecoder.getHeight();

imageWidth = mDecoder.getWidth();

} catch (Exception e) {

e.printStackTrace();

}

}

private class OnScrollLisrener extends RecyclerView.OnScrollListener {

@Override

public void onScrolled(RecyclerView recyclerView, int dx, int dy) {

super.onScrolled(recyclerView, dx, dy);

*//获取广告的itemView*

View ggView = linearLayoutManager.findViewByPosition(ggPosition);

if (ggView == null) {

return;

}

if (ggImageView == null) {

return;

}

*//获取滑动控件的高*

parentHeight = mRecyclerView.getHeight();

*//图片距离滑动控件的上下距离*

int topOrBottomPadding;

int top = ggView.getTop();

int left = 0;

int right = imageWidth;

int bottom = ggView.getBottom();

*//如果图片比滑动控件短*

if (parentHeight > imageHeight) {

topOrBottomPadding = (parentHeight - imageHeight) / 2;

*//获取item的高*

int itemHeight = ggView.getHeight();

if (top >= parentHeight - itemHeight - topOrBottomPadding) {

*//如果超出底部,就一直显示图片的底部*

bottom = imageHeight;

top = bottom - itemHeight;

} else if (top <= topOrBottomPadding) {

*//如果超出顶部,就一直显示图片的顶部*

top = 0;

bottom = top + itemHeight;

} else {

*//处于图片中的时候,自由滑动*

top -= topOrBottomPadding;

bottom = top + itemHeight;

}

}

mRect.set(left, top, right, bottom);

*//异步（异步时会卡...貌似是因为线程延迟的问题）*

*// executorService.execute(bitmapRunnable);*

*//同步→*

ggBitmap = mDecoder.decodeRegion(mRect, null);

if (ggBitmap != null) {

ggImageView.setImageBitmap(ggBitmap);

}

*//←同步*

}

}

*//添加假数据*

private void addJia() {

List<ItemBean> list = new ArrayList<>();

for (int i = 0; i < 12; i++) {

list.add(new ItemBean(ItemBean.ITEM\_PT, "aaa" + i));

}

list.add(new ItemBean(ItemBean.ITEM\_GG, "ggg"));

for (int i = 0; i < 12; i++) {

list.add(new ItemBean(ItemBean.ITEM\_PT, "bbb" + i));

}

adapter.setData(list);

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
* 158
* 159
* 160
* 161
* 162
* 163
* 164
* 165
* 166

**- adapter代码**

import android.support.v7.widget.RecyclerView;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ImageView;

import java.util.List;

/\*\*

\* Created by 郭辉 on 2017/12/18 16:37.

\* TODO:

\*/

public class CopyZhiHuAdapter extends RecyclerView.Adapter<CopyZhiHuAdapter.ViewHolder> {

private List<ItemBean> mList = null;

private MainActivity mainActivity = null;

public CopyZhiHuAdapter(MainActivity mainActivity) {

this.mainActivity = mainActivity;

}

public void setData(List<ItemBean> list) {

this.mList = list;

notifyDataSetChanged();

}

@Override

public int getItemViewType(int position) {

ItemBean itemBean = mList.get(position);

return itemBean.getType();

}

@Override

public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {

ViewHolder holder = null;

if (viewType == ItemBean.ITEM\_PT) {

holder = new PtViewHolder(LayoutInflater.from(parent.getContext()).inflate(R.layout.item\_layout\_pt, parent, false));

} else if (viewType == ItemBean.ITEM\_GG) {

holder = new GgViewHolder(LayoutInflater.from(parent.getContext()).inflate(R.layout.item\_layout\_gg, parent, false));

}

return holder;

}

@Override

public void onBindViewHolder(ViewHolder holder, int position) {

int itemViewType = getItemViewType(position);

switch (itemViewType) {

case ItemBean.ITEM\_GG:

GgViewHolder ggHolder = (GgViewHolder) holder;

mainActivity.setGGViewPosition(position, ggHolder.ggView);

break;

case ItemBean.ITEM\_PT:

break;

}

}

@Override

public int getItemCount() {

return mList != null ? mList.size() : 0;

}

class ViewHolder extends RecyclerView.ViewHolder {

public ViewHolder(View itemView) {

super(itemView);

}

}

/\*\*

\* 普通

\*/

class PtViewHolder extends ViewHolder {

public PtViewHolder(View itemView) {

super(itemView);

}

}

/\*\*

\* 广告

\*/

class GgViewHolder extends ViewHolder {

ImageView ggView;

View itemView;

public GgViewHolder(View itemView) {

super(itemView);

this.itemView = itemView;

ggView = itemView.findViewById(R.id.item\_imageview);

}

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96

-普通item布局:item\_layout\_pt

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#ffffff"

android:orientation="vertical">

<View

android:layout\_width="match\_parent"

android:layout\_height="1px"

android:background="#dddddd" />

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:gravity="center"

android:padding="20dp"

android:text="普通的itemView" />

<View

android:layout\_width="match\_parent"

android:layout\_height="1px"

android:background="#dddddd" />

</LinearLayout>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 广告item布局:item\_layout\_gg

<?xml version="1.0" encoding="utf-8"?>

<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical">

<ImageView

android:id="@+id/item\_imageview"

android:layout\_width="0dp"

android:layout\_height="0dp"

android:src="@mipmap/default\_loading"

app:layout\_constraintDimensionRatio="W,1.5:3"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

app:layout\_constraintTop\_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* ItemBean

/\*\*

\* Created by 郭辉 on 2017/12/18 16:49.

\* TODO:

\*/

public class ItemBean {

public static final int ITEM\_PT = 0;

public static final int ITEM\_GG = 1;

public ItemBean(int type, String name) {

this.type = type;

this.name = name;

}

private int type;

private String name;

public int getType() {

return type;

}

public void setType(int type) {

this.type = type;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34

**-项目地址：**

链接：<https://pan.baidu.com/s/1bOonaY> 密码：g1k8