

RecyclerView ItemDecoration 完全解析

星火燎原16 [关注](#)

0.764 2018.12.23 16:24:55 字数 1,398 阅读 3,863

我们都知道，使用 RecyclerView 时，我们不能像 ListView 那样通过 `setDivider()` 的方式来设置分割线，好在 Android 为我们提供了定制性更强的 ItemDecoration 来为 RecyclerView 设置分割线。

什么是 ItemDecoration ?

顾名思义 ItemDecoration 就是 Item 的装饰，我们可以在 Item 的上下左右添加自定义的装饰，比如 横线，图案。同时系统已经为我们提供了一个 `DividerItemDecoration`，如果这个 `DividerItemDecoration` 不满足我们的需求，我们就可以通过自定义 ItemDecoration 来实现了。

下面我们看下系统的 `DividerItemDecoration`：

引入 `DividerItemDecoration`(系统提供)

`DividerItemDecoration` 的使用非常简单，只需添加下面代码即可：

```
1 | DividerItemDecoration decoration = new DividerItemDecoration(this,DividerItemDecoratio  
2 | recyclerView.addItemDecoration(decoration);
```

效果：



DividerItemDecoration.png

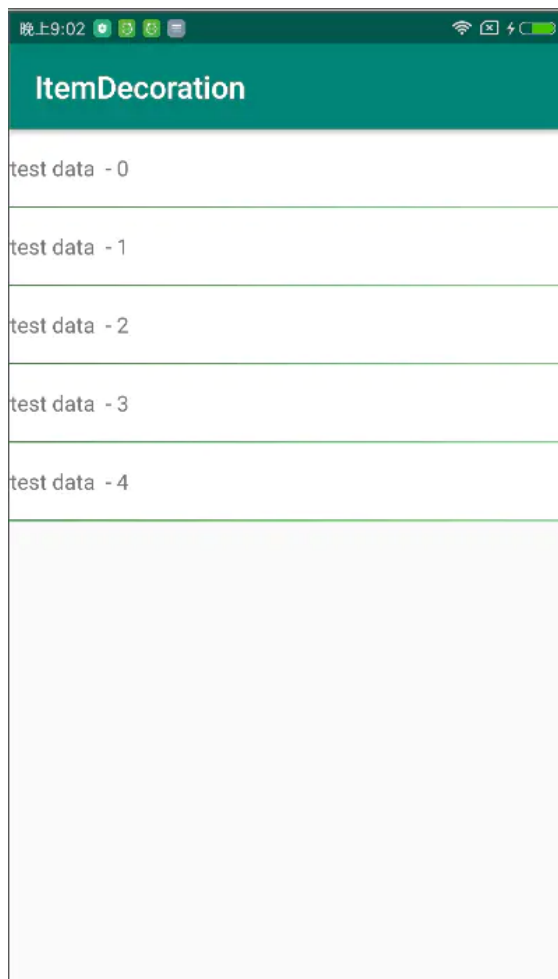
如果想要修改 DividerItemDecoration 的颜色和高度，可以调用它的 `setDrawable(drawable)` 设置一个 Drawable 对象

```

1 // MainActivity.java
2 DividerItemDecoration decoration = new DividerItemDecoration(this, DividerItemDecorati
3 Drawable dividerDrawable = getResources().getDrawable(R.drawable.drawable_divider);
4 decoration.setDrawable(dividerDrawable);
5 recyclerView.addItemDecoration(decoration);
6
7 // res/drawable/drawable_divider.xml
8 <?xml version="1.0" encoding="utf-8"?>
9 <shape xmlns:android="http://schemas.android.com/apk/res/android">
10     android:shape="rectangle">
11     <!-- 渐变色 -->
12     <gradient
13         android:angle="135"
14         android:centerColor="#4CAF50"
15         android:endColor="#2E7D32"
16         android:startColor="#81C784"
17         android:type="linear" />
18
19     <size android:height="1dp" />
20 </shape>

```

效果:



dividerItemDecoration-drawable.png

自定义 ItemDecoration

自定义 ItemDecoration，主要需要重写以下三个方法：

```

1  /**
2   * 自定义 ItemDecoration
3   */
4  public class LinearItemDecoration extends RecyclerView.ItemDecoration {
5
6      @Override
7      public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView.State state) {
8          super.getItemOffsets(outRect, view, parent, state);
9      }
10
11     @Override
12     public void onDraw(@NonNull Canvas c, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
13         super.onDraw(c, parent, state);
14     }
15
16     @Override
17     public void onDrawOver(@NonNull Canvas c, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
18         super.onDrawOver(c, parent, state);
19     }

```

1. getItemOffsets()

getItemOffsets() 主要作用是在 item 的四周留下边距，效果和 margin 类似，item 的四周留下边距后，我们就可以通过 onDraw() 在这个边距上绘制了。

```

1 | public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView.State state) {

```

(1) 参数 Rect outRect : 表示 item 的上下左右所留下的边距。其中 outRect 的 left, top, right, bottom 即为 item 四周留下的边距的距离，默认都为 0；示意图如下：

(2) 参数 View view : 指当前 item 的 View 对象；

(3) 参数 RecyclerView parent : 指 RecyclerView 本身；

(4) RecyclerView.State state : 指 RecyclerView 当前的状态；

1.1 getItemOffsets() 应用例子：

既然 getItemOffsets(Rect outRect) 方法可以设置 item 四周的边距大小，那就可以设置 recyclerview 背景色和 item 四周的边距，使得 item 四周的边距透出 recyclerview 背景色来达到分割线的目的。

当然 item 的背景色需要和 recyclerview 的背景色不一致才有效果；

首先将 recyclerview 的背景色设置为 colorAccent 红色，将 item 的背景色设置为 白色：

```

1  <!-- activity_main.xml ----->
2  <?xml version="1.0" encoding="utf-8"?>
3  <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".MainActivity">
9
10     <android.support.v7.widget.RecyclerView
11         android:id="@+id/rv_recycler"
12         android:layout_width="match_parent"
13         android:layout_height="match_parent"
14         android:background="@color/colorAccent" />
15
16 </android.support.constraint.ConstraintLayout>

1  <!-- item_recycler.xml ----->
2  <?xml version="1.0" encoding="utf-8"?>
3  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="50dp"

```

```

6      android:background="#fff"
7      android:gravity="center_vertical"
8      android:orientation="vertical">
9
10     <TextView
11         android:id="@+id/tv_title"
12         android:layout_width="match_parent"
13         android:layout_height="40dp"
14         android:gravity="center_vertical" />
15
16 </LinearLayout>

```

然后继承 ItemDecoration 类，重写 getOffsets() 方法，将 outRect 的上边距 top 设置为 10px;

```

1  /**
2   * 自定义 ItemDecoration
3   */
4  public class LinearItemDecoration extends RecyclerView.ItemDecoration {
5
6      @Override
7      public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView.State state) {
8          super.getItemOffsets(outRect, view, parent, state);
9          outRect.top = 10; // 10px
10     }
11 }

```

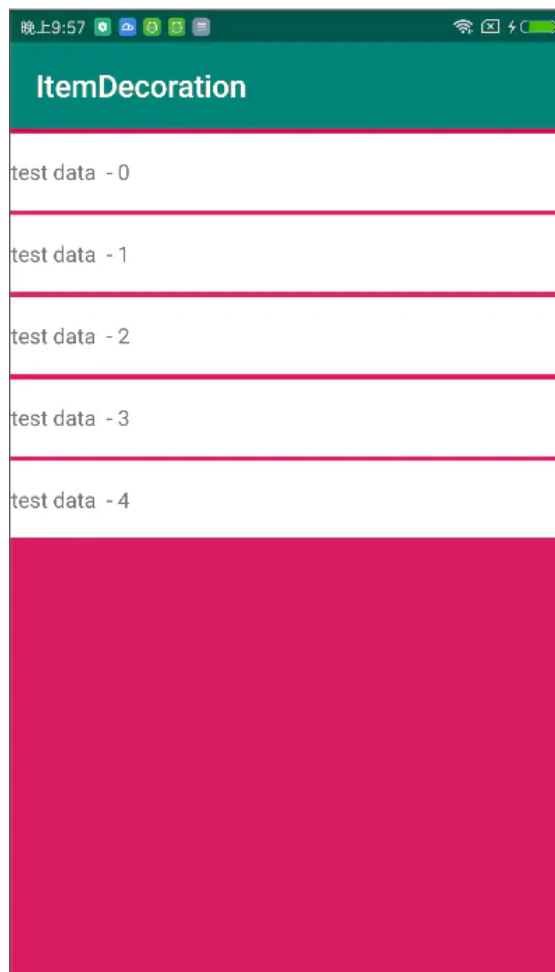
最后将这个 LinearItemDecoration 添加到 RecyclerView 中：

```

1  LinearItemDecoration decoration = new LinearItemDecoration();
2  recyclerView.addItemDecoration(decoration);

```

效果如下(下方的红色是因为RecyclerView 高度为 match_parent,但 item 数据只有 5 条)：



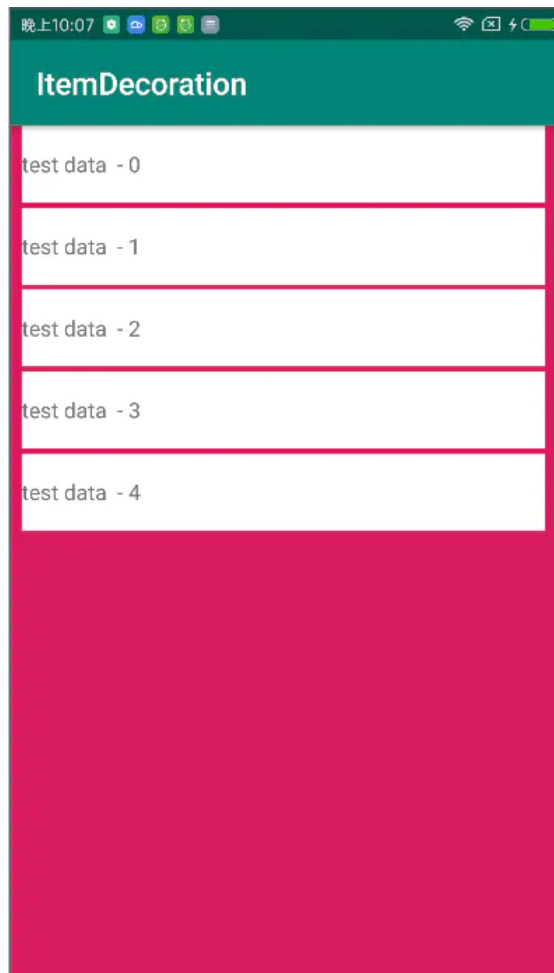
linearItemDecoration.png

从效果图可以看出：每一个 item 的 顶部都有一个红色的背景线，包括第一个 item 顶部也有 (怎么解决呢？见 2.1 节详解)，

同理，我们可以设置为 底部 10px,左侧 20px ,右侧 40px；

```
1 public class LinearItemDecoration extends RecyclerView.ItemDecoration {  
2  
3     @Override  
4     public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {  
5         super.getItemOffsets(outRect, view, parent, state);  
6         outRect.bottom = 10;  
7         outRect.left = 20;  
8         outRect.right = 40;  
9     }  
10 }
```

效果：



linearItemDecoration-bottom-left-right.png

可以看到：每个 item 的左侧，底部，右侧都有了间距，露出了 RecyclerView 的背景色了。

2. onDraw()

```
1 @Override  
2 public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {  
3     super.onDraw(canvas, parent, state);  
4 }
```

onDraw() 函数中的 parent , state 参数和 getItemOffsets() 方法中的参数含义是一样的，
canvas 参数是 getItemOffsets() 函数所留下的左右上下的空白区域对应的 Canvas 画布对象。
我们可以在这个区域中利用 Paint 画笔绘制任何图形。

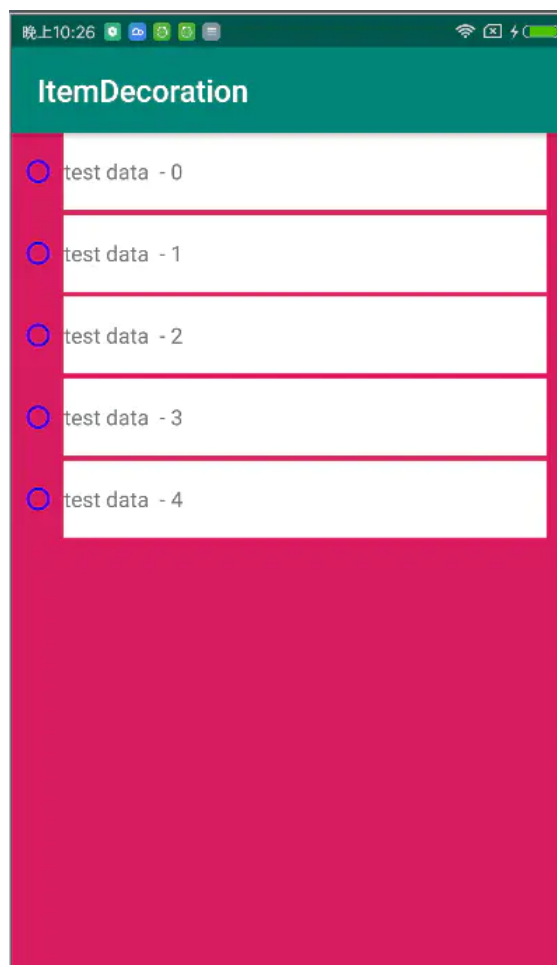
比如在 item 左侧绘制一个 空心圆。

```

1  /**
2   * 自定义 ItemDecoration
3   */
4  public class LinearItemDecoration extends RecyclerView.ItemDecoration {
5
6      private static final String TAG = "LinearItemDecoration";
7      private Paint paint;
8
9      public LinearItemDecoration() {
10         paint = new Paint(Paint.ANTI_ALIAS_FLAG);
11         paint.setColor(Color.BLUE);
12         paint.setStyle(Paint.Style.STROKE);
13         paint.setStrokeWidth(5);
14     }
15
16     @Override
17     public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull Recyc
18         super.getItemOffsets(outRect, view, parent, state);
19         Log.e(TAG, "getItemOffsets: " );
20         outRect.bottom = 10;
21         outRect.left = 100;
22         outRect.right = 40;
23     }
24
25     @Override
26     public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull Re
27         super.onDraw(canvas, parent, state);
28         Log.e(TAG, "onDraw: ");
29         for (int i = 0; i < parent.getChildCount(); i++) {
30             View childView = parent.getChildAt(i);
31             canvas.drawCircle(50, childView.getTop() + childView.getHeight() / 2, 20, pa
32         }
33     }
34 }
35 }

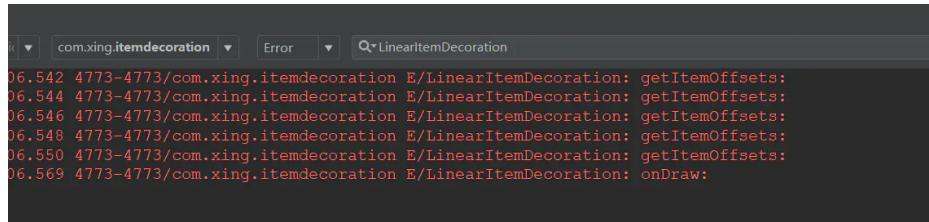
```

效果：



onDraw-circle.png

我们在 `getItemOffsets()` 和 `onDraw()` 方法中都添加了日志，日志打印如下：



log.png

从日志中可以看出：`getItemOffsets()` 方法执行了 5 遍，和数据源个数是一样的，但 `onDraw()` 方法只执行了一遍，由此我们知道，`getItemOffsets()` 是针对每个 item 都会执行一次，也就是说每个 item 的 `outRect` 可以设置为不同值，但是 `onDraw()`, `onDrawOver()` 是针对 `ItemDecoration` 的，不是针对 item 的，只执行一次。所以我们在 `onDraw()`, `onDrawOver()` 中绘制的时候，需要遍历每个 item 进行绘制。

优雅获取 `outRect` 中的值

在上面例子中，我们在 `onDraw` 中获取 `outRect` 中的值都是写成计算好的固定值，显然这种硬编码的方式不利于扩展，其实，我们可以通过 `LayoutManager` 来获取 `getItemOffsets()` 中设置的 `outRect` 的值。

```
1  @Override
2  public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.LayoutManager layoutManager) {
3      super.onDraw(canvas, parent, state);
4      Log.e(TAG, "onDraw: ");
5      RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
6      for (int i = 0; i < parent.getChildCount(); i++) {
7          View childView = parent.getChildAt(i);
8          int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
9          int topDecorationHeight = layoutManager.getTopDecorationHeight(childView);
10         int rightDecorationWidth = layoutManager.getRightDecorationWidth(childView);
11         int bottomDecorationHeight = layoutManager.getBottomDecorationHeight(childView);
12     }
13 }
```

上面硬编码可以改成：

```
1  @Override
2  public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.LayoutManager layoutManager) {
3      super.onDraw(canvas, parent, state);
4      Log.e(TAG, "onDraw: ");
5      RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
6      for (int i = 0; i < parent.getChildCount(); i++) {
7          View childView = parent.getChildAt(i);
8          int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
9          int left = leftDecorationWidth / 2;
10         canvas.drawCircle(left, childView.getTop() + childView.getHeight() / 2, 20, paint);
11     }
12 }
```

2.1 扩展1 -- 减少背景设置，避免过度绘制

为了减少过度绘制，我们将 `activity_main` 中 `RecyclerView`, `item_recycler` 中的背景全部去掉，不设置任何背景，然后在 `LinearItemDecoration` 中进行纯绘制分割线，代码如下：

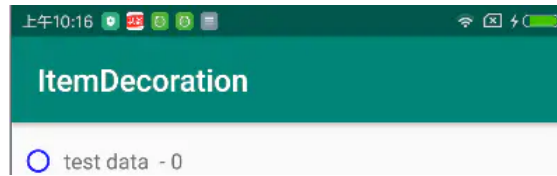
```
1  public class LinearItemDecoration extends RecyclerView.ItemDecoration {
2
3      private static final String TAG = "LinearItemDecoration";
4      private Paint paint;
5      private Paint dividerPaint;
6  }
```

```

7 public LinearItemDecoration() {
8     paint = new Paint(Paint.ANTI_ALIAS_FLAG);
9     paint.setColor(Color.BLUE);
10    paint.setStyle(Paint.Style.STROKE);
11    paint.setStrokeWidth(5);
12
13    dividerPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
14    dividerPaint.setColor(Color.parseColor("#e6e6e6"));
15    dividerPaint.setStyle(Paint.Style.FILL);
16 }
17
18 @Override
19 public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView.LayoutManager layoutManager, @NonNull State state) {
20     super.getItemOffsets(outRect, view, parent, state);
21     Log.e("ItemOffsets", "getItemOffsets: ");
22     outRect.bottom = 5;
23     outRect.left = 100;
24 }
25
26 @Override
27 public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.LayoutManager layoutManager, @NonNull State state) {
28     super.onDraw(canvas, parent, state);
29     Log.e(TAG, "onDraw: ");
30     RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
31     for (int i = 0; i < parent.getChildCount(); i++) {
32         View childView = parent.getChildAt(i);
33         int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
34         int topDecorationHeight = layoutManager.getTopDecorationHeight(childView);
35         int rightDecorationWidth = layoutManager.getRightDecorationWidth(childView);
36         int bottomDecorationHeight = layoutManager.getBottomDecorationHeight(childView);
37         int left = leftDecorationWidth / 2;
38         canvas.drawCircle(left, childView.getTop() + childView.getHeight() / 2, 20);
39         // getItemOffsets()中的设置的是 bottom = 5px;所以在 drawRect 时, top 为 childView.getBottom() - 5;
40         canvas.drawRect(new Rect(
41             leftDecorationWidth,
42             childView.getBottom(),
43             childView.getWidth() + leftDecorationWidth,
44             childView.getBottom() + bottomDecorationHeight
45         ), dividerPaint);
46     }
47 }
48
49 @Override
50 public void onDrawOver(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.LayoutManager layoutManager, @NonNull State state) {
51     super.onDrawOver(canvas, parent, state);
52 }
53 }

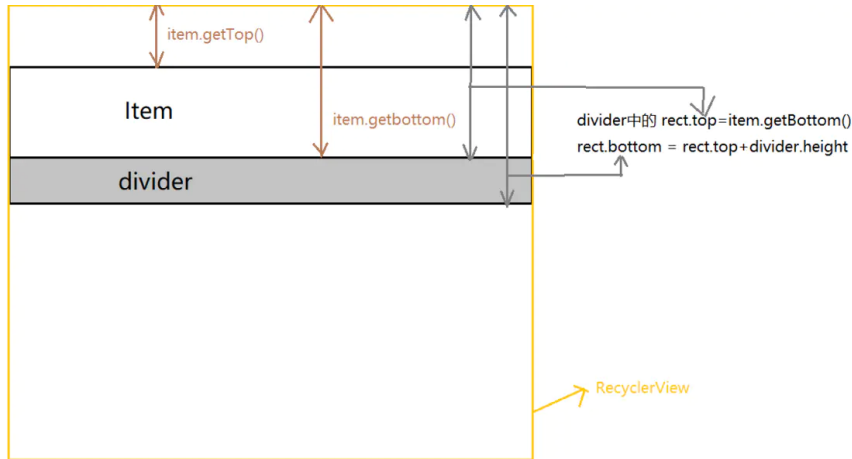
```

效果:



divider-draw.png

注意：上面 getItemOffsets() 中设置的是 bottom = 5px; 所以在 onDraw() 方法的 drawRect 时, top值为 childView.getBottom(), bottom值为 top+bottomDecorationHeight。



divider-top-bottom.png

同理：getItemOffsets() 中设置是 top = 5px, 那么在 onDraw() 方法 drawRect 时, bottom 值为 childView.getTop(), top 值为 bottom - topDecorationHeight

2.2 扩展2 -- 实现竖直进度分割线

在 getItemOffsets() 方法中左侧留下空白区域, 然后在 onDraw() 方法中绘制 圆和 竖线。代码如下：

```

1  <!-- item_recycler.xml ---->
2  <?xml version="1.0" encoding="utf-8"?>
3  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="wrap_content"
7      android:background="@android:color/white"
8      android:gravity="center_vertical"
9      android:orientation="vertical"
10     android:padding="10dp">
11
12     <TextView
13         android:id="@+id/tv_title"
14         android:layout_width="match_parent"
15         android:layout_height="wrap_content"
16         android:textColor="#332"
17         android:textSize="16sp"
18         tools:text="我是 title" />
19
20     <TextView
21         android:id="@+id/tv_time"
22         android:layout_width="match_parent"
23         android:layout_height="wrap_content"
24         android:layout_marginTop="5dp"
25         android:gravity="center_vertical"
26         android:textColor="#666"
27         android:textSize="14sp"
28         tools:text="woshi" />
29
30 </LinearLayout>
31
32 <!-- activity_main.xml --->
33 <?xml version="1.0" encoding="utf-8"?>

```

```

34 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com
35     xmlns:app="http://schemas.android.com/apk/res-auto"
36     xmlns:tools="http://schemas.android.com/tools"
37     android:layout_width="match_parent"
38     android:layout_height="match_parent"
39     tools:context=".MainActivity">
40
41     <android.support.v7.widget.RecyclerView
42         android:id="@+id/rv_recycler"
43         android:layout_width="match_parent"
44         android:layout_height="match_parent"
45         android:background="#dfdfdf" />
46
47 </android.support.constraint.ConstraintLayout>

```

```

1 // ProgressItemDecoration.java
2 public class ProgressItemDecoration extends RecyclerView.ItemDecoration {
3
4     private Context context;
5     private Paint circlePaint;
6     private Paint linePaint;
7     private int radius;
8     private int curPosition = 0; // 当前进行中的位置
9
10    public ProgressItemDecoration(Context context) {
11        this.context = context;
12        circlePaint = new Paint(Paint.ANTI_ALIAS_FLAG);
13        circlePaint.setColor(context.getResources().getColor(R.color.colorPrimary));
14        circlePaint.setStyle(Paint.Style.FILL);
15        radius = dp2Px(8);
16        circlePaint.setStrokeWidth(dp2Px(2));
17
18        linePaint = new Paint(Paint.ANTI_ALIAS_FLAG);
19        linePaint.setColor(context.getResources().getColor(R.color.colorPrimary));
20        linePaint.setStrokeWidth(dp2Px(2));
21    }
22
23    @Override
24    public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
25        super.getItemOffsets(outRect, view, parent, state);
26        outRect.top = dp2Px(20);
27        outRect.left = dp2Px(50);
28        outRect.right = dp2Px(20);
29    }
30
31    @Override
32    public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
33        super.onDraw(canvas, parent, state);
34        int childCount = parent.getChildCount();
35        RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
36        for (int i = 0; i < childCount; i++) {
37            View childView = parent.getChildAt(i);
38            int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
39            int topDecorationHeight = layoutManager.getTopDecorationHeight(childView);
40            // 获取当前 item 是 recyclerview 的第几个 childview
41            int childLayoutPosition = parent.getChildLayoutPosition(childView);
42            float startX = leftDecorationWidth / 2;
43            float stopX = startX;
44            // 圆顶部部分竖线, 起点 Y
45            float topStartY = childView.getTop() - topDecorationHeight;
46            // 圆顶部部分竖线, 终点 Y
47            float topStopY = childView.getTop() + childView.getHeight() / 2 - radius;
48
49            // 圆底部部分竖线, 起点 Y
50            float bottomStartY = childView.getTop() + childView.getHeight() / 2 + radius;
51            // 圆底部部分竖线, 终点 Y
52            float bottomStopY = childView.getBottom();
53
54            // 位置超过 curPosition 时, 竖线颜色设置为浅色
55            if (childLayoutPosition > curPosition) {
56                linePaint.setColor(context.getResources().getColor(R.color.colorPrimaryLight));
57                circlePaint.setColor(context.getResources().getColor(R.color.colorPrimaryLight));
58            } else {
59                linePaint.setColor(context.getResources().getColor(R.color.colorPrimary));
60                circlePaint.setColor(context.getResources().getColor(R.color.colorPrimary));
61            }
62            circlePaint.setStyle(Paint.Style.STROKE);
63            linePaint.setStyle(Paint.Style.FILL);
64        }
65    }
66
67    // 绘制圆

```

```

67         if (childLayoutPosition == curPosition) {
68             circlePaint.setStyle(Paint.Style.STROKE);
69             canvas.drawCircle(leftDecorationWidth / 2, childView.getTop() + childV
70         }
71         canvas.drawCircle(leftDecorationWidth / 2, childView.getTop() + childView.
72
73         // 绘制竖线，第 0 位置上只需绘制 下半部分
74         if (childLayoutPosition == 0) {
75             // 当前 item position = curPosition 时，绘制下半部分竖线时，颜色设置为浅色
76             if (childLayoutPosition == curPosition) {
77                 linePaint.setColor(context.getResources().getColor(R.color.colorPr
78             }
79             canvas.drawLine(startX, bottomStartY, startX, bottomStopY, linePaint);
80             // 最后位置上，只需绘制上半部分
81         } else if (childLayoutPosition == parent.getAdapter().getItemCount() - 1)
82             canvas.drawLine(startX, topStartY, startX, topStopY, linePaint);
83         } else {
84             // 都要绘制
85             canvas.drawLine(startX, topStartY, startX, topStopY, linePaint);
86             // 当前 item position = curPosition 时，绘制下半部分竖线时，颜色设置为浅色
87             if (childLayoutPosition == curPosition) {
88                 linePaint.setColor(context.getResources().getColor(R.color.colorPr
89             }
90             canvas.drawLine(startX, bottomStartY, startX, bottomStopY, linePaint);
91         }
92     }
93
94 }
95
96 /**
97  * 设置进行中的位置
98  *
99  * @param recyclerView
100  * @param position
101  */
102 public void setDoingPosition(RecyclerView recyclerView, int position) {
103     if (recyclerView == null) {
104         throw new IllegalArgumentException("RecyclerView can't be null");
105     }
106     if (recyclerView.getAdapter() == null) {
107         throw new IllegalArgumentException("RecyclerView Adapter can't be null");
108     }
109     if (position < 0) {
110         throw new IllegalArgumentException("position can't be less than 0");
111     }
112     recyclerView.getLayoutManager().getItemCount();
113     if (position > recyclerView.getAdapter().getItemCount() - 1) {
114         throw new IllegalArgumentException("position can't be greater than item co
115     }
116     this.curPosition = position;
117 }
118
119 private int dp2Px(int value) {
120     return (int) TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP, value, con
121 }
122 }
123
124

```

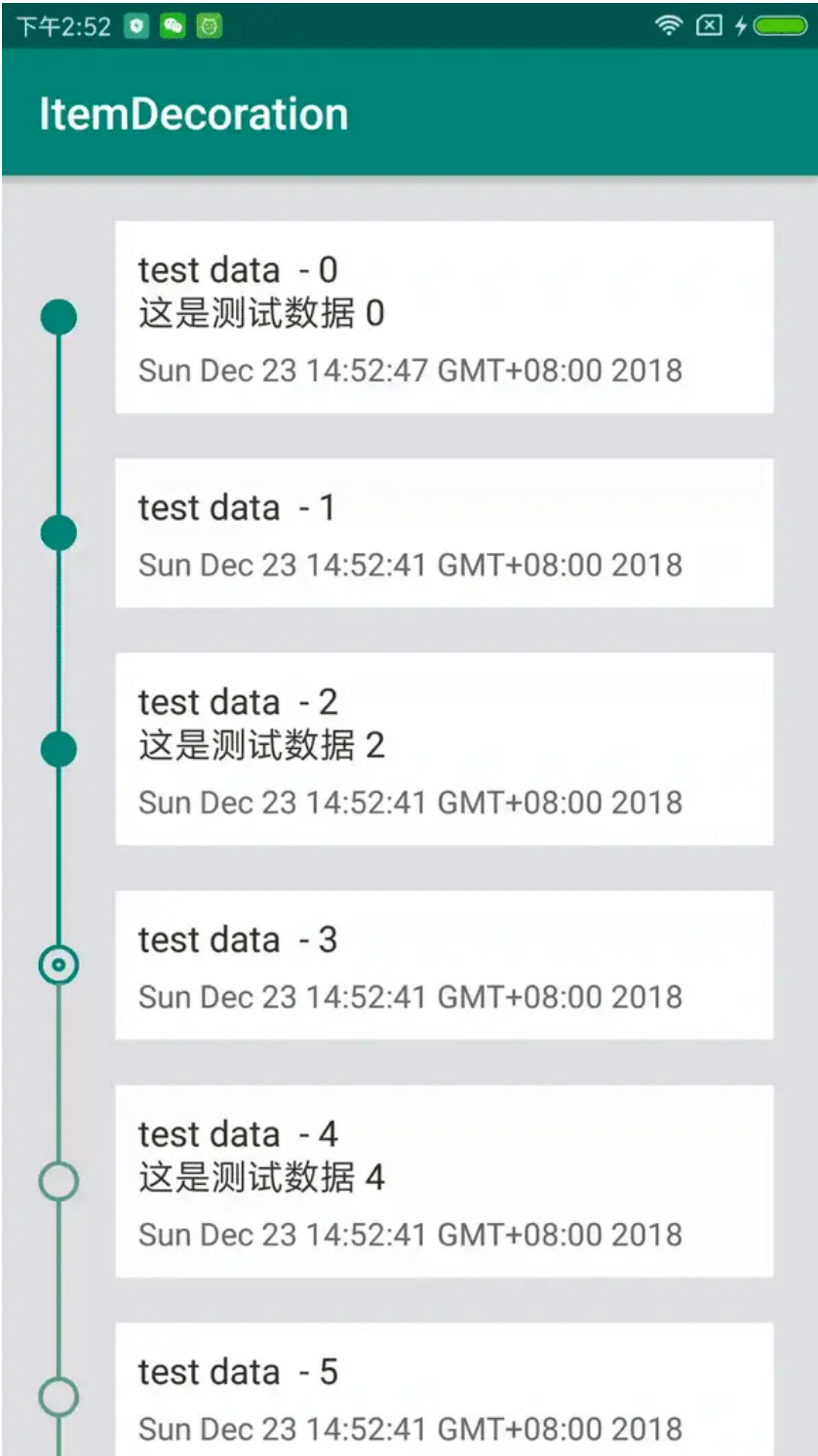
```

1 // MainActivity.java
2 public class MainActivity extends AppCompatActivity {
3
4     @Override
5     protected void onCreate(Bundle savedInstanceState) {
6         super.onCreate(savedInstanceState);
7         setContentView(R.layout.activity_main);
8         RecyclerView recyclerView = findViewById(R.id.rv_recycler);
9         recyclerView.setLayoutManager(new LinearLayoutManager(this, LinearLayoutManager
10         List<String> list = new ArrayList<>();
11         for (int i = 0; i < 10; i++) {
12             String data = "test data - " + i;
13             if (i % 2 == 0) {
14                 data = data + "\n" + "这是测试数据 " + i;
15             }
16             list.add(data);
17         }
18         BaseRecyclerViewAdapter adapter = new BaseRecyclerViewAdapter<String>(list, R.layout.i
19
20         @Override
21         protected void bind(BaseRecyclerViewAdapter<String> adapter, BaseViewHolder ho
22             holder.setText(R.id.tv_title, data);
23             holder.setText(R.id.tv_time, new Date().toString());
24     }
25 }

```

```
24     };
25     recyclerView.setAdapter(adapter);
26     ProgressItemDecoration decoration = new ProgressItemDecoration(this);
27     decoration.setDoingPosition(recyclerView, 0);
28     recyclerView.addItemDecoration(decoration);
29 }
30 }
31 }
```

效果：



vertical-progress.gif

3. onDrawOver()

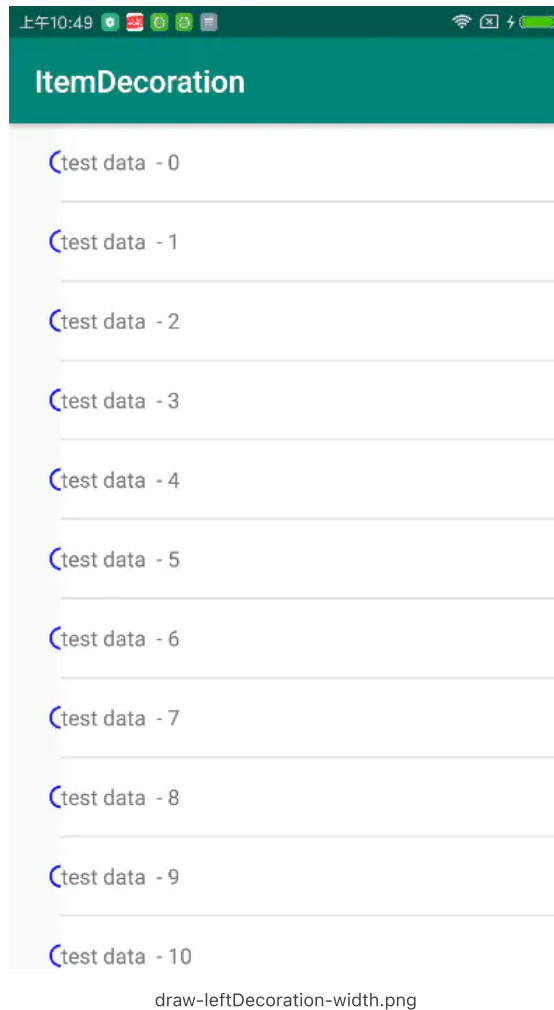
当我们将上面例子中绘制左侧的空心圆的位置改为：圆心 x 坐标为 leftDecorationWidth，同时将 item 背景色设置为 白色：

```

1  @Override
2  public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull RecyclerView.State state) {
3      super.getItemOffsets(outRect, view, parent, state);
4      Log.e("ItemOffsets", "getItemOffsets: ");
5      outRect.bottom = 5;
6      outRect.left = 100;
7  }
8
9  @Override
10 public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull RecyclerView.State state) {
11     super.onDraw(canvas, parent, state);
12     Log.e(TAG, "onDraw: ");
13     RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
14     for (int i = 0; i < parent.getChildCount(); i++) {
15         View childView = parent.getChildAt(i);
16         int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
17         int bottomDecorationHeight = layoutManager.getBottomDecorationHeight(childView);
18         int left = leftDecorationWidth / 2;
19         // canvas.drawCircle(left, childView.getTop() + childView.getHeight() / 2, leftDecorationWidth / 2);
20         canvas.drawCircle(leftDecorationWidth, childView.getTop() + childView.getHeight() / 2, leftDecorationWidth / 2);
21         // getItemOffsets()中的设置的是 bottom = 5px;所以在 drawRect 时, top 为 childView.getTop() + bottomDecorationHeight
22         canvas.drawRect(new Rect(
23             leftDecorationWidth,
24             childView.getBottom(),
25             childView.getWidth() + leftDecorationWidth,
26             childView.getBottom() + bottomDecorationHeight
27         ), dividerPaint);
28     }
29 }

```

效果：



我们发现：绘制的空心圆被 item 遮挡了右边部分，变为不可见了，这是因为在这个绘制的流程中，先调用 ItemDecoration 的 onDraw() 方法，然后再调用 item 的 onDraw() 方法，最后再调用 ItemDecoration 的 onDrawOver() 方法。

因此，当我们想要在 item 的绘制显示一些内容时，将绘制的逻辑写在 onDrawOver() 方法即可。

下面我们实现在 item 与 左侧 decoration 交汇处绘制一个 apple icon ,并将 item 中的内容文本设置为居中显示，代码如下：

```

1  public class LinearItemDecoration extends RecyclerView.ItemDecoration {
2
3      private static final String TAG = "LinearItemDecoration";
4      private Paint paint;
5      private Paint dividerPaint;
6      private Bitmap iconBitmap;
7
8      public LinearItemDecoration(Context context) {
9          paint = new Paint(Paint.ANTI_ALIAS_FLAG);
10         paint.setColor(Color.BLUE);
11         paint.setStyle(Paint.Style.STROKE);
12         paint.setStrokeWidth(5);
13
14         dividerPaint = new Paint(Paint.ANTI_ALIAS_FLAG);
15         dividerPaint.setColor(Color.parseColor("#e6e6e6"));
16         dividerPaint.setStyle(Paint.Style.FILL);
17
18         iconBitmap = BitmapFactory.decodeResource(context.getResources(), R.drawable.i
19     }
20
21     @Override
22     public void getItemOffsets(@NonNull Rect outRect, @NonNull View view, @NonNull Rec
23         super.getItemOffsets(outRect, view, parent, state);
24         Log.e("ItemOffsets", "getItemOffsets: ");
25         outRect.bottom = 5;
26         outRect.left = 100;
27     }
28
29     @Override
30     public void onDraw(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNull I
31         super.onDraw(canvas, parent, state);
32         Log.e(TAG, "onDraw: ");
33         RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
34         for (int i = 0; i < parent.getChildCount(); i++) {
35             View childView = parent.getChildAt(i);
36             int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
37             int bottomDecorationHeight = layoutManager.getBottomDecorationHeight(child
38             int left = leftDecorationWidth / 2;
39             // canvas.drawCircle(left, childView.getTop() + childView.getHeight() / 2,
40             // canvas.drawCircle(leftDecorationWidth, childView.getTop() + childView.ge
41
42             // getItemOffsets()中的设置的是 bottom = 5px;所以在 drawRect 时, top 为 childV
43             canvas.drawRect(new Rect(
44                 leftDecorationWidth,
45                 childView.getBottom(),
46                 childView.getWidth() + leftDecorationWidth,
47                 childView.getBottom() + bottomDecorationHeight
48             ), dividerPaint);
49         }
50     }
51
52     @Override
53     public void onDrawOver(@NonNull Canvas canvas, @NonNull RecyclerView parent, @NonNi
54         super.onDrawOver(canvas, parent, state);
55         RecyclerView.LayoutManager layoutManager = parent.getLayoutManager();
56         for (int i = 0; i < parent.getChildCount(); i++) {
57             View childView = parent.getChildAt(i);
58             int leftDecorationWidth = layoutManager.getLeftDecorationWidth(childView);
59             canvas.drawBitmap(iconBitmap, leftDecorationWidth - iconBitmap.getWidth() ,
60                 childView.getTop() + childView.getHeight() / 2 - iconBitmap.getHei
61         }
62     }
63 }

```

效果：

