背景是这样的：有一个需求要对下方的关注按钮实现如下动画，动画的最后要根据滑动位置对关注按钮进行缩放，缩放结束时整体大小为控件原始大小的90%，最终效果图如下所示（模拟器是 4.2 的系统，最上面的沉浸式有点问题，忽略之）：

如图，关注按钮向上滑动到某临界位置时，按钮需要缩小；同样，向下滑动到相同的临界位置时，需要放大。

在布局文件中，关注按钮整体布局为RelativeLayout，爱心ImageView和文字TextView居中。

由于动画需要根据滑动位置动态计算，因此不能用缩放动画实现。当滑动到临界位置时，我做如下处理：

float scale = 1- 0.1f \* (scrollY - 100) / 10; //scale范围：1.0~0.9

mLayoutFollow.setPivotX(1.0f);

mLayoutFollow.setPivotY(1.0f);

mLayoutFollow.setScaleX(scale);

mLayoutFollow.setScaleY(scale);

1

2

3

4

5

上述代码中，缩放比例scale 的值是根据滑动距离scrollY 动态计算出来的，其取值范围为1.0~0.9；设置setPivotX(1.0f)和setPivotY(1.0f)的目的，是想让关注按钮（RelativeLayout）的缩放轴点为右下角，但是悲剧的是，这样的设置并没有生效，缩放的中心始终为左上角。这与 xml 中设置PivotX和PivotY属性是相矛盾的（在 xml中，都设置为0.0f，则缩放中心为控件左上角；都设置为0.5f，则缩放中心为控件中心；都设置为1.0f，则缩放中心为控件右下角，详见博文 图解 Android 动画中 android:pivotX 和 android:pivotY 属性的含义）。

测试后发现，只要 setPivotX(float x) 或 setPivotY(float y) 中有一个值为 0，最后的结果都是缩放中心为左上角；而当不设置这两个值时（即去掉setPivotX/Y这两行代码），缩放中心为控件中心。（奇怪，为什么呢？。。）

想到 xml 中还可以将PivotX属性设置为整数，于是这样设置：

mLayoutFollow.setPivotX(100);

mLayoutFollow.setPivotY(0.0f);

1

2

发现，缩放中心变为控件最上方，但偏右一些，说明setPivotX(100)生效了。

据此启发，于是，得到最终设置缩放中心为控件右下角的方法如下：

float scale = 1- 0.1f \* (scrollY - 100) / 10;//scale范围：1.0~0.9

mLayoutFollow.setPivotX(X); //X为控件宽度的px值，以实际情况为准

mLayoutFollow.setPivotY(Y); //Y为控件宽度的px值，以实际情况为准

mLayoutFollow.setScaleX(scale);

mLayoutFollow.setScaleY(scale);

1

2

3

4

5

设置控件缩放中心为其他位置的方法亦类似。

查看 View 的setPivotX源码如下：

/\*\*

\* Sets the x location of the point around which the view is

\* {@link #setRotation(float) rotated} and {@link #setScaleX(float) scaled}.

\* By default, the pivot point is centered on the object.

\* Setting this property disables this behavior and causes the view to use only the

\* explicitly set pivotX and pivotY values.

\*

\* @param pivotX The x location of the pivot point.

\* @see #getRotation()

\* @see #getScaleX()

\* @see #getScaleY()

\* @see #getPivotY()

\*

\* @attr ref android.R.styleable#View\_transformPivotX

\*/

public void setPivotX(float pivotX) {

if (!mRenderNode.isPivotExplicitlySet() || pivotX != getPivotX()) {

invalidateViewProperty(true, false);

mRenderNode.setPivotX(pivotX);

invalidateViewProperty(false, true);

invalidateParentIfNeededAndWasQuickRejected();

}

}

1

24

翻译前面的注释：默认情况下（不设置时），轴点位于控件中心处；设置该属性后，控件只会明确地使用所设置的轴点位置。而参数pivotX的含义是轴点的 x 位置（这里应该就是绝对位置了）。

不知道这样的理解是不是正确的。。。╮(╯▽╰)╭ 尴尬~~~抛砖引玉，还望知道的小伙伴告知~

---------------------

作者：hust\_twj

来源：CSDN

原文：https://blog.csdn.net/hust\_twj/article/details/78877087

版权声明：本文为博主原创文章，转载请附上博文链接！