移动应用开发



简单UI实践实验报告

|  |  |
| --- | --- |
| 学 院： | 计算机与信息技术学院 |
| 专 业： | 计算机科学与技术 |
| 班 级： | 计科1602 |
| 姓 名： | 麻锦涛 |
| 学 号： | 16281262 |

2019年 5月 20日

1. 功能介绍

“CustomGallery”是一款实现3D相册，实现了女星的图片在一定空间的滑动跳转，以及其倾斜、放大和倒影处理的效果。

1. 具体实现
2. Activity

**import** android.app.Activity;  
**import** android.os.Bundle;  
  
**public class** MainActivity **extends** Activity {  
 */\*\* Called when the activity is first created. \*/* @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
  
 setContentView(R.layout.main);  
  
 **int**[] images = { R.drawable.photo1, R.drawable.photo2,  
 R.drawable.photo3, R.drawable.photo4, R.drawable.photo5,  
 R.drawable.photo6, R.drawable.photo7, R.drawable.photo8, };  
  
 ImageAdapter adapter = **new** ImageAdapter(**this**, images);  
 adapter.createReflectedImages();  
  
 GalleryFlow galleryFlow = (GalleryFlow) findViewById(R.id.gallery\_flow);  
 galleryFlow.setAdapter(adapter);  
  
 }  
}

1. View

<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"** >  
<**com.android.CustomGallery.GalleryFlow  
 android:id="@+id/gallery\_flow"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"** />  
</**LinearLayout**>

1. 特色之处

（1）重载Gallery类

需要加入倒影和3D切换的效果，因此需要重载Gallery类，其中有两个方法我们需要重写，一个是onSizeChanged(),另外一个是getChildStaticTransformation()。下面我们看看onSizeChanged()需要做的事情。

1. @Override
2. protected void onSizeChanged(int w, int h, int oldw, int oldh)
3. {//重写计算旋转的中心
4. mCoveflowCenter = getCenterOfCoverflow();
5. super.onSizeChanged(w, h, oldw, oldh);
6. }

上面主要做的事情就是在改变大小的时候，重新计算滑动切换时需要旋转变化的中心。下面计算图片位置时，会重新计算。

1. protected boolean getChildStaticTransformation(View child, Transformation trans)
2. { //图像的中心点和宽度
3. final int childCenter = getCenterOfView(child);
4. final int childWidth = child.getWidth();
5. int rotationAngle = 0;
7. trans.clear();
8. trans.setTransformationType(Transformation.TYPE\_BOTH);        // alpha 和 matrix 都变换
10. if (childCenter == mCoveflowCenter)
11. {      // 正中间的childView
12. transformImageBitmap((ImageView) child, trans, 0);
13. }
14. else
15. {          // 两侧的childView
16. rotationAngle = (int) ( ( (float) (mCoveflowCenter - childCenter) / childWidth ) \* mMaxRotationAngle );
17. if (Math.abs(rotationAngle) > mMaxRotationAngle)
18. {
19. rotationAngle = (rotationAngle < 0) ? -mMaxRotationAngle : mMaxRotationAngle;
20. }　//根据偏移角度对图片进行处理，看上去有3D的效果。
21. transformImageBitmap((ImageView) child, trans, rotationAngle);
22. }
24. return true;
25. }

上面就是重载Gallery的时候，主要就是做图形变化，效果图里面的图片出现了倾斜就是这里处理的结果，目的就是让照片的人看上去有立体感。

2、Adapter适配器

这里使用很多控件都涉及适配器。这里呢，适配器就是用来绑定数据源和目标控件的一个中间件。我们需要重载BaseAdapter作为我们Gallery的适配器。主要是处理一下源图像，加入倒影，生成新的数据源图片。

1. public boolean createReflectedForAdapter()
2. {
3. final int reflectionGap = 4;
4. final int Height = 200;
5. int index = 0;
6. for (Map<String, Object> map : list)
7. {
8. Integer id = (Integer) map.get("image");
9. // 获取原始图片
10. Bitmap originalImage = BitmapFactory.decodeResource(mContext.getResources(), id);
11. int width = originalImage.getWidth();
12. int height = originalImage.getHeight();
13. float scale = Height / (float)height;
15. Matrix sMatrix = new Matrix();
16. sMatrix.postScale(scale, scale);
17. Bitmap miniBitmap = Bitmap.createBitmap(originalImage, 0, 0,
18. originalImage.getWidth(), originalImage.getHeight(), sMatrix, true);
20. //是否原图片数据，节省内存
21. originalImage.recycle();
23. int mwidth = miniBitmap.getWidth();
24. int mheight = miniBitmap.getHeight();
25. Matrix matrix = new Matrix();
26. // 图片矩阵变换（从低部向顶部的倒影）
27. matrix.preScale(1, -1);
28. // 截取原图下半部分
29. Bitmap reflectionImage = Bitmap.createBitmap(miniBitmap, 0, mheight/2, mwidth, mheight/2, matrix, false);
30. // 创建倒影图片（高度为原图3/2）
31. Bitmap bitmapWithReflection = Bitmap.createBitmap(mwidth, (mheight + mheight / 2), Config.ARGB\_8888);
32. // 绘制倒影图（原图 + 间距 + 倒影）
33. Canvas canvas = new Canvas(bitmapWithReflection);
34. // 绘制原图
35. canvas.drawBitmap(miniBitmap, 0, 0, null);
36. Paint paint = new Paint();
37. // 绘制原图与倒影的间距
38. canvas.drawRect(0, mheight, mwidth, mheight + reflectionGap, paint);
39. // 绘制倒影图
40. canvas.drawBitmap(reflectionImage, 0, mheight + reflectionGap, null);
42. paint = new Paint();
43. // 线性渐变效果
44. LinearGradient shader = new LinearGradient(0, miniBitmap.getHeight(), 0, bitmapWithReflection.getHeight()
45. + reflectionGap, 0x70ffffff, 0x00ffffff, TileMode.CLAMP);
46. paint.setShader(shader);
47. // 倒影遮罩效果
48. paint.setXfermode(new PorterDuffXfermode(Mode.DST\_IN));
49. // 绘制倒影的阴影效果
50. canvas.drawRect(0, mheight, mwidth, bitmapWithReflection.getHeight() + reflectionGap, paint);
51. ImageView imageView = new ImageView(mContext);  // 设置倒影图片
52. imageView.setImageBitmap(bitmapWithReflection);
53. imageView.setLayoutParams(new GalleryView.LayoutParams((int)(width \* scale),
54. (int)(mheight \* 3 / 2.0 + reflectionGap)));
55. imageView.setScaleType(ScaleType.MATRIX);
56. mImages[index++] = imageView;
57. }
58. return true;
59. }

上面是一个[图片处理](https://cloud.tencent.com/product/ci?idx=1)过程，主要做的事情就是生成倒影，而效果图里面底下是有倒影的。我们在适配器添加图片的时候，会把适配器原生图片进行处理，加入倒影的效果。

具体图片滑动的过程，Gallery会处理好，我们要做的事情其实就是提供添加了特效的图片数据源，以及处理3D显示的变化效果，最后都会提供View作为显示图像给Gallery用来显示。

1. 效果展示

主要实现了相册的3D跳转效果，这里以本人最爱的女星苍井空小姐姐为例：

 