[Android] 使用Matrix矩阵类对图像进行缩放、旋转、对比度、亮度处理

前一篇文章讲述了Android拍照、截图、保存并显示在ImageView控件中,该篇文章继续讲述Android图像处理技术,主要操作包括:通过打开相册里的图片,使用Matrix对图像进行缩放、旋转、移动、对比度、亮度、饱和度操作,希望对大家有所帮助.

一. 显示打开图片

首先,设置activity_main.xml布局如下所示:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    tools:context="com.example.cangeimagetest.MainActivity"
    tools:ignore="MergeRootFrame" >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical" >
            <Button
                android:id="@+id/button1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="选择图片" />
            <TextView
                android:id="@+id/textView1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:visibility="invisible"
                android:text="原图显示" />
            <ImageView
                android:id="@+id/imageView1"
                android:layout_width="wrap_content"
                android:layout_gravity="center_horizontal"
                android:layout_height="wrap_content" />
            <TextView
                android:id="@+id/textView2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:visibility="invisible"
                android:text="变化后的图片" />
            <ImageView
                android:id="@+id/imageView2"
                android:layout_gravity="center_horizontal"
                android:layout_marginBottom="20dp"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" />
    </LinearLayout>
```

```
         <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_alignParentBottom="true" >
        <Button
                android:id="@+id/button2"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="缩小" />
        <Button
                android:id="@+id/button3"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="放大" />
        <Button
                android:id="@+id/button4"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="旋转" />
        <Button
                android:id="@+id/button5"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="饱和" />
        <Button
                android:id="@+id/button6"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:text="对比" />
    </LinearLayout>
</RelativeLayout>
```

然后,在Mainctivity.java中public class MainActivity extends Activity函数添加代码如下:

```
private Button selectBn;
private ImageView imageShow;
private ImageView imageCreate;
private TextView textview1;
private TextView textview2;
private Bitmap bmp; //原始图片

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    selectBn = (Button) findViewById(R.id.button1);
    imageShow = (ImageView) findViewById(R.id.imageView1);
    imageCreate = (ImageView) findViewById(R.id.imageView2);
    textview1 = (TextView) findViewById(R.id.textView1);
    textview2 = (TextView) findViewById(R.id.textView2);

    //选择图片
    selectBn.setOnClickListener(new OnClickListener() {
```

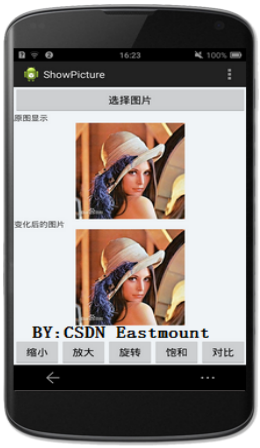
```
        @Override
                        public void onClick(View v) {
                Intent intent = new Intent(Intent.ACTION_PICK,
                                android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
                startActivityForResult(intent, 0 );
        }
    });
    if (savedInstanceState == null) {
        getFragmentManager().beginTransaction()
                .add(R.id.container, new PlaceholderFragment())
                .commit();
    }
}
//显示两张图片
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if(resultCode==RESULT_OK) {
                ShowPhotoByImageView(data);     //显示照片
                CreatePhotoByImageView();          //创建图片
        }
}
```

再调用自定义函数实现显示图片：

```
//自定义函数 显示打开的照片在ImageView1中
public void ShowPhotoByImageView(Intent data) {
        Uri imageFileUri = data.getData();
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);
        int width = dm.widthPixels;    //手机屏幕水平分辨率
        int height = dm.heightPixels;  //手机屏幕垂直分辨率
        Log.v("height", ""+height );
        Log.v("width", ""+width);
        try {
                // Load up the image's dimensions not the image itself
                BitmapFactory.Options bmpFactoryOptions = new BitmapFactory.Options();
                bmpFactoryOptions.inJustDecodeBounds = true;
                bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri), null,
bmpFactoryOptions);
                int heightRatio = (int)Math.ceil(bmpFactoryOptions.outHeight/(float)height);
                int widthRatio = (int)Math.ceil(bmpFactoryOptions.outWidth/(float)width);
                Log.v("bmpheight", ""+bmpFactoryOptions.outHeight);
                Log.v("bmpheight", ""+bmpFactoryOptions.outWidth);
                if(heightRatio>1&&widthRatio>1) {                        if(heightRatio>widthRatio) {
                                bmpFactoryOptions.inSampleSize = heightRatio*2;
                        }
                        else {
                                bmpFactoryOptions.inSampleSize = widthRatio*2;
                        }
                }
                 //图像真正解码
            bmpFactoryOptions.inJustDecodeBounds = false;
            bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri), null,
bmpFactoryOptions);
            imageShow.setImageBitmap(bmp); //将剪裁后照片显示出来
            textview1.setVisibility(View.VISIBLE);         } catch(FileNotFoundException e) {
                e.printStackTrace();
        }
}
//创建第二张图片并显示
```

```
public void CreatePhotoByImageView() {
                                        try {
            Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp); //画布 传入位图用于绘制
        Paint paint = new Paint(); //画刷 改变颜色 对比度等属性
        canvas.drawBitmap(bmp, 0, 0, paint);    //错误:没有图片 因为参数bmp写成createBmp
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
        } catch(Exception e) {
                e.printStackTrace();
        }
}
```

显示的效果如下图所示,该图叫莱娜图(Lenna),是图像处理中经常使用的样例图.



二. Matrix操作

然后通过Matrix对图像进行处理操作,在onCreate函数中添加点击事件:

```
//缩小图片
Button button2=(Button)findViewById(R.id.button2);
button2.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                SmallPicture();
        }
});
//放大图片
    Button button3=(Button)findViewById(R.id.button3);
    button3.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View v) {
                        BigPicture();
                }
        });
   //旋转图片
Button button4=(Button)findViewById(R.id.button4);
button4.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                TurnPicture();
        }
});
//图片饱和度改变
```

```
Button button5=(Button)findViewById(R.id.button5); button5.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                SaturationPicture();
        }
});
//图片对比度改变
Button button6=(Button)findViewById(R.id.button6);
button6.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                ContrastPicture();
        }
});
```

最后分别自定义函数各操作实现,代码如下:

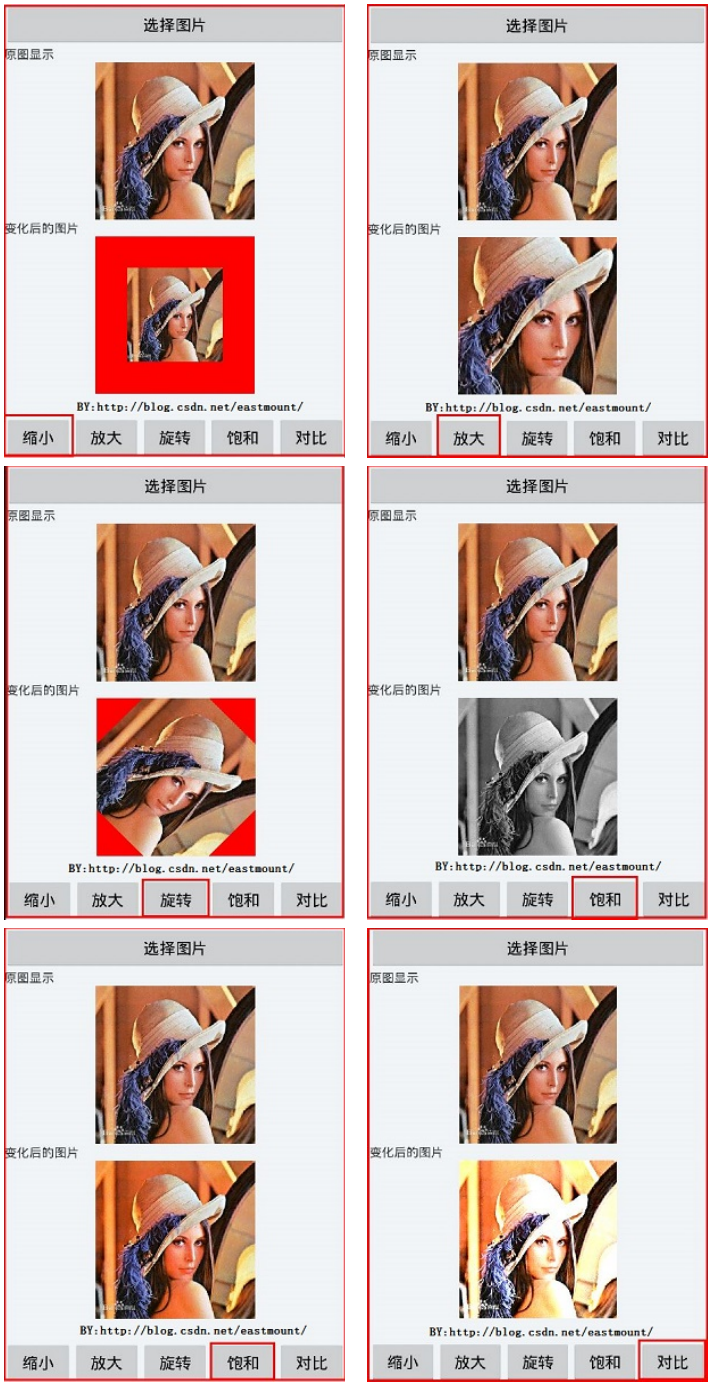
```
//缩小图片
private void SmallPicture() {
        Matrix matrix = new Matrix();
        //缩放区间 0.5-1.0
        if(smallbig>0.5f)
                smallbig=smallbig-0.1f;
        else
                smallbig=0.5f;
        //x y坐标同时缩放
        matrix.setScale(smallbig,smallbig,bmp.getWidth()/2,bmp.getHeight()/2);
        Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp); //画布 传入位图用于绘制
        Paint paint = new Paint(); //画刷 改变颜色 对比度等属性
        canvas.drawBitmap(bmp, matrix, paint);
        imageCreate.setBackgroundColor(Color.RED);
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
    }
  //放大图片
private void BigPicture() {
        Matrix matrix = new Matrix();
        //缩放区间 0.5-1.0
        if(smallbig<1.5f)
                smallbig=smallbig+0.1f;
        else
                smallbig=1.5f;
        //x y坐标同时缩放
        matrix.setScale(smallbig,smallbig,bmp.getWidth()/2,bmp.getHeight()/2);
        Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp);
        Paint paint = new Paint();
        canvas.drawBitmap(bmp, matrix, paint);
        imageCreate.setBackgroundColor(Color.RED);
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
}
//旋转图片
private void TurnPicture() {
        Matrix matrix = new Matrix();
        turnRotate=turnRotate+15;
        //选择角度 饶(0,0)点选择 正数顺时针 负数逆时针 中心旋转
        matrix.setRotate(turnRotate,bmp.getWidth()/2,bmp.getHeight()/2);
```

```
        Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp);          Paint paint = new Paint();
        canvas.drawBitmap(bmp, matrix, paint);
        imageCreate.setBackgroundColor(Color.RED);
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
}
//改变图像饱和度
private void SaturationPicture() {
        //设置饱和度 0表示灰度图像 大于1饱和度增加 0-1饱和度减小
        ColorMatrix cm = new ColorMatrix();
        cm.setSaturation(saturation);
        Paint paint = new Paint();
        paint.setColorFilter(new ColorMatrixColorFilter(cm));
        //显示图片
        Matrix matrix = new Matrix();
        Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp);
        canvas.drawBitmap(bmp, matrix, paint);
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
        saturation=saturation+0.1f;
        if(saturation>=1.5f) {
                saturation=0f;
        }
}
//设置图片对比度
private void ContrastPicture() {
        ColorMatrix cm = new ColorMatrix();
        float brightness = -25;  //亮度
        float contrast = 2;        //对比度
        cm.set(new float[] {
                contrast, 0, 0, 0, brightness,
                0, contrast, 0, 0, brightness,
                0, 0, contrast, 0, brightness,
                0, 0, 0, contrast, 0
        });
        Paint paint = new Paint();
        paint.setColorFilter(new ColorMatrixColorFilter(cm));
        //显示图片
        Matrix matrix = new Matrix();
        Bitmap createBmp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas canvas = new Canvas(createBmp);
        canvas.drawBitmap(bmp, matrix, paint);
        imageCreate.setImageBitmap(createBmp);
        textview2.setVisibility(View.VISIBLE);
}
```

同时自定义变量如下：

```
//图片变换参数
private float smallbig=1.0f;   //缩放比例
private int turnRotate=0;       //旋转度数
private float saturation=0f;    //饱和度
```

它的运行结果如下图所示：



需要指出的是：该项目仅仅讲述处理的过程,并没有考虑很多因素,如:有的图像显示可能超出屏幕,没有载入图片点击处理按钮报错,横竖屏切换导致不显示图片,最下面按钮可能被遮挡,图像放大画布没有变,因为为认为显示一张改变后的图片效果更好,而该工程仅仅是对比.图像缩放移动触屏变换更好,下一篇讲述.

XML布局推荐：http://www.apkbus.com/forum.php?mod=viewthread&tid=44949

解决画布跟着图片放大：http://www.eoeandroid.com/thread-3162-1-1.html

三．Matrix处理的原理

Android中可以通过Matrix和ColorMatrix对图像进行处理.

1.Matrix

图像空间变换,包括旋转、剪裁、缩放或移动.Matrix类中每个数字都将应用于图像上每个点的3个坐标x\y\z之一.

如下代码通过setValues设置值.(1,0,0)表示x坐标转换x=1x+0y+0z,同样y=0x+1y+0z,z=0x+0y+1z.该矩阵不做任何变换.如果第一行改为(.5f,0,0),那么图像在x轴上将图像压缩50%.移动见setTranslate()函数.

```
Matrix matrix = new Matrix();
matrix.setValues(new float[] {
        1, 0, 0,
        0, 1, 0,
        0, 0, 1
});
```

2.ColorMatrix

在Canvas(画布)对象上绘制时既可使用Matrix方法,也可使用ColorMatrix来改变在Canvas对象上绘制的Paint(画刷)对象.对图像的像素处理时,每个像素由RGBA值组成(Red Green Blue Alpha).具体方法推荐博文:http://www.cnblogs.com/leon19870907/articles/1978065.html

最后希望该文章对大家有所帮助,尤其是Android初学者.该文章是讲述Android使用Matrix处理图片的基础文章,如果有不足或错误地方,请见谅~参考资料《Android多媒体开发高级编程 著: Shawn Van Every》

下载地址:http://download.csdn.net/detail/eastmount/8082043

(By:Eastmount 2014-10-26 夜2点 http://blog.csdn.net/eastmount)