[Android] 触屏setOnTouchListener实现图片缩放、移动、绘制和添加水印

前一篇文章讲述了Android实现图片Matrix矩阵类缩放、旋转、对比度、亮度、饱和度处理,但是真正的图片软件都是使用触屏实现图片缩放、移动、添加水印等功能,所以该篇文章主要通过setOnTouchListener监听实现该功能.希望文章对大家有所帮助.

一.图片缩放实现

首先先简单介绍Android如何实现触屏缩放图片和移动图片,新建TouchImageViw工程.设计XML中activity_main.xml布局:

```
<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="match_parent"
    tools:context="com.example.touchimagetest.MainActivity"
    tools:ignore="MergeRootFrame" >
     <!-- 顶部路径 -->
    <RelativeLayout
        android:id="@+id/MyLayout_top"
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="40dp"
        android:layout_alignParentTop="true"
        android:gravity="center">
        <TextView
                android:id="@+id/textView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textSize="15sp"
                android:text="显示图片路径" />
    </RelativeLayout>
    <!-- 底部按钮 -->
    <RelativeLayout
        android:id="@+id/MyLayout_bottom"
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:layout_alignParentBottom="true"
        android:gravity="center">
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="horizontal"
                android:layout_alignParentBottom="true" >
                <Button
                        android:id="@+id/button1"
                        android:layout_width="wrap_content"
                        android:layout_height="match_parent"
```

```
                    android:layout_weight="1"                                    android:text="打开" />
            </LinearLayout>
        </RelativeLayout>
        <!-- 显示图片 -->
    <RelativeLayout
        android:id="@+id/Content_Layout"
        android:orientation="horizontal"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_above="@id/MyLayout_bottom"
        android:layout_below="@id/MyLayout_top"
        android:background="#EFDFDF"
        android:gravity="center">
        <ImageView
                android:id="@+id/imageView1"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:layout_gravity="center_horizontal"
                        android:scaleType="matrix" />
    </RelativeLayout>
</RelativeLayout>
```

需要注意的是按钮中设置其android:layout_weight="1"属性,因为后面会增加至4个按钮,都设置为1表示每个宽度占水平布局的1/4比例.(图见后)该布局是我非常喜欢的布局,通过相对布局RelativeLayout设置相应的上中下对应图片路径、图片、按钮.

然后,在Mainctivity.java中public class MainActivity extends Activity函数添加代码如下,添加点击按钮监听事件:

```
//自定义变量
private Button openImageBn;           //打开图片
private Bitmap bmp;                   //原始图片
private TextView pathText;            //路径TextView
private String path;                  //存储图片路径
private ImageView imageShow;          //显示图片
private final int IMAGE_CODE = 0;     //打开图片
//触屏缩放图片
private static final int NONE = 0;     //初始状态
private static final int DRAG = 1;     //拖动
private static final int ZOOM = 2;     //缩放
private int mode = NONE;               //当前事件
private float oldDist;
private PointF startPoint = new PointF();
private PointF middlePoint = new PointF();
private Matrix matrix = new Matrix();
private Matrix savedMatrix = new Matrix();

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    openImageBn = (Button) findViewById(R.id.button1);
    pathText = (TextView) findViewById(R.id.textView1);
    imageShow = (ImageView) findViewById(R.id.imageView1);
    //打开图片
    openImageBn.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, IMAGE_CODE);         }
    });
}
```

添加intent意图startActivityForResult对应的onActivityResult函数实现打开图片：

```
//打开图片
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
         super.onActivityResult(requestCode, resultCode, data);
            if(resultCode==RESULT_OK && requestCode==IMAGE_CODE) {
                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
                try {
                    //载入图片尺寸大小没载入图片本身 true
                    BitmapFactory.Options bmpFactoryOptions = new BitmapFactory.Options();
                    bmpFactoryOptions.inJustDecodeBounds = true;
                    bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri),
null, bmpFactoryOptions);
                //outHeight图像高 outWidth图像宽
                    int heightRatio = (int)Math.ceil(bmpFactoryOptions.outHeight/(float)height);
                    int widthRatio = (int)Math.ceil(bmpFactoryOptions.outWidth/(float)width);
                    //inSampleSize表示图片占原图比例 =1表示原图
                    if(heightRatio>1&&widthRatio>1) {
                        if(heightRatio>widthRatio) {
                            bmpFactoryOptions.inSampleSize = heightRatio;                        }
                        else {
                            bmpFactoryOptions.inSampleSize = widthRatio;
                        }
                    }
                //图像真正解码 false
                    bmpFactoryOptions.inJustDecodeBounds = false;
                    bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri),
null, bmpFactoryOptions);
                    imageShow.setImageBitmap(bmp);                    //显示文件路径
                    String[] filePathColumn= {MediaStore.Images.Media.DATA};
                    Cursor cursor = getContentResolver().query(imageFileUri, filePathColumn, null, null,
null);
                    cursor.moveToFirst(); //将光标移至开头
                    int columnIndex = cursor.getColumnIndex(filePathColumn[0]); //获得用户选择图片的索引
                path = cursor.getString(columnIndex);                    cursor.close();
                    pathText.setText("path="+path);

                } catch(FileNotFoundException e) {
                    e.printStackTrace();
                }
            }  //end if
}
```

此时可以实现打开图片,显示的效果如下图所示：

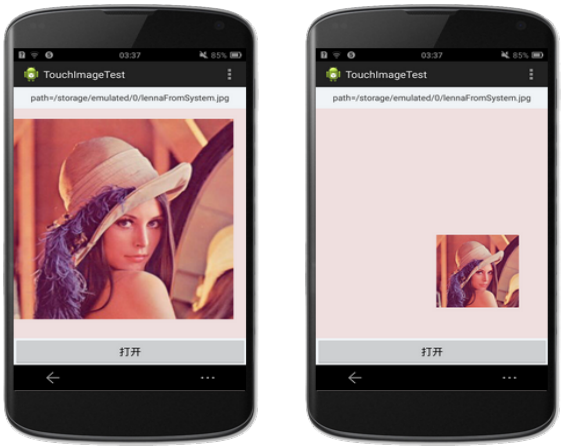
然后在onCreate()函数中openImageBn.setOnClickListener打开图片后添加如下代码,通过setOnTouchListener监听实现图片缩放移动功能.PS:这部分网上资料较多,也比较相似.作者参照《Android多媒体开发高级编程》.

```
//触屏缩放图片监听 注:XML中修改android:scaleType="matrix"
imageShow.setOnTouchListener(new OnTouchListener() {

        @Override
        public boolean onTouch(View v, MotionEvent event) {
                ImageView view = (ImageView) v;
                switch (event.getAction() & MotionEvent.ACTION_MASK) {
                case MotionEvent.ACTION_DOWN: //手指按下
                        savedMatrix.set(matrix);
                        startPoint.set(event.getX(), event.getY());
                        mode = DRAG;
                        break;
                case MotionEvent.ACTION_UP:
                case MotionEvent.ACTION_POINTER_UP:
                        mode = NONE;
                        break;
                case MotionEvent.ACTION_POINTER_DOWN:
                        oldDist = spacing(event); //如果两点距离大于10 多点模式
                        if (oldDist > 10f) {
                                savedMatrix.set(matrix);
                                midPoint(middlePoint, event);
                                mode = ZOOM;
                        }
                        break;
                case MotionEvent.ACTION_MOVE:
                        if (mode == DRAG) { //拖动
                                matrix.set(savedMatrix);
                                matrix.postTranslate(event.getX() - startPoint.x, event.getY() -
startPoint.y);
                        } else if (mode == ZOOM) { //缩放
                                float newDist = spacing(event);
                                if (newDist > 10f) {
                                        matrix.set(savedMatrix);
                                        float scale = newDist / oldDist;
                                        matrix.postScale(scale, scale, middlePoint.x, middlePoint.y);
                                }                       }
                        break;
                } //end switch
                view.setImageMatrix(matrix);
                return true;
        }
        //两点距离
```

```
        private float spacing(MotionEvent event) {                float x = event.getX(0) - event.getX(1);
                float y = event.getY(0) - event.getY(1);
                return FloatMath.sqrt(x * x + y * y);
        }
        //两点中点
        private void midPoint(PointF point, MotionEvent event) {
                float x = event.getX(0) + event.getX(1);
                float y = event.getY(0) + event.getY(1);
                point.set(x / 2, y / 2);
        }
});
```

运行结果如下图所示：



下面讲讲作者遇到的几个问题及需要注意的几个地方：

1.如何使用RelativeLayout进行布局，上面布局非常优秀，是我参照了郑海波学长一个代码例子的布局，个人觉得比较喜欢.

2.在图像缩放的时候需要设置ImageView的属性android:scaleType="matrix",这里涉及到前一篇讲述的Matrix矩阵类的知识,但是使用该类后图像如何居中呢?

设置其父布局为android:gravity="center"是不行的,作者查阅很多资料,有两种方法：一种是设置ImageView的scaleType属性为center,在缩放时代码设置为matrix,感觉不太好;另一种是通过下面的代码可以实现居中显示,但是在缩放时需要注意设置其图片左上角显示位置,否则会出现跳动到(0,0)现象.

```
//设置图片居中 起点=未使用屏幕/2=(屏幕分辨率-图片宽高)/2
int widthCenter=imageShow.getWidth()/2-bmp.getWidth()/2;
int heightCenter=imageShow.getHeight()/2-bmp.getHeight()/2;
Matrix matrix = new Matrix();
matrix.postTranslate(widthCenter, heightCenter);
imageShow.setImageMatrix(matrix);
imageShow.setImageBitmap(alteredBitmap);
```

希望高手能提供更好的方法,因为跳动现象,我自己也会继续去学习.

3.图片ImageView设置了setOnTouchListener监听,能实现缩放图片,那么如果还点击一个按钮"文字"后实现的是拖放文字框,点击"绘制"是在图像上绘制图片,怎么实现呢?点击一个按钮怎样实现一个触摸监听呢?

由于这个很难查阅资料,像美图秀秀哪样的功能,所以我自己只能想到通过修改一个变量赋值判断触屏监听执行那个方法,自定义函数private int flagOnTouch=0,当其1-显示图片,2-添加文字,3-缩放图片,4-画图.在点击按钮事件中对其进行赋值,在setOnTouchListener中判断.(希望提供更好的方法)

4.缩放图片时最好设置最小的边界,当缩小当一定范围尺寸固定不变,放大最好也有越界等处理.

二.原理介绍

下面讲述触摸事件的一些基础知识,方便巩固上面的工程.

Android许多UI元素都继承View类,该类支持触摸,该方法是setOnTouchListener,其接受一个实现OnTouchListener接口对象作为参数.每当触摸ImageView调用活动中的onTouch方法,再通过查看传递到onTouch方法中的MotionEvent对象,可以确定发生哪种类型的触摸,可调用getAction方法:

MotionEvent.ACTION_DOWN:表明视图已经接收一次触摸,按下时触发

MotionEvent.ACTION_UP:表明视图停止接受一次触摸,被放开时触发

MotionEvent.ACTION_POINTER_DOWN:当屏幕上已经有一点被按住,再按下其他点时触发

MotionEvent.ACTION_POINTER_UP:当屏幕上有多个点被按住,松开其中一个点时触发(即非最后一个点被放开时)

MotionEvent.ACTION_MOVE:表明在发生一次触摸之后,在ACTION_UP之前发生了某种移动

MotionEvent.ACTION_CANCEL:表明触摸已经被取消,因此该忽略它

同时可以调用MotionEvent对象上的getX和getY来确定触摸事件发生的位置.

如上面的代码,当ACTION_DOWN手指第一次按下时设置mode=DRAG,当ACTION_POINTER_DOWN再按下一个手指时判断距离大于10设置mode=ZOOM,当ACTION_MOVE移动时判断两指为缩放,一指为移动即可实现.在通过新坐标和老坐标设置图像变换.参考《Android多媒体开发高级编程》

(PS:作者才接触Android一个星期,说起来很简单啊!但做起来还是非常难的~)

三.工程实现详解

上面简单的缩放功能与基本原理(干货)都讲述了,读者可以自己去实现功能,下面就如何实现触摸setOnTouchListener图片缩放、移动、绘制和添加水印.都是基于上面代码修改实现的.需要注意的是在图片修改时需要创建一个新的Bitmap alteredBitmap(bmp图片)、Canvas canvas(画布)、Paint paint(画刷)实现.

1.图片缩放和图片绘制

首先,为XML布局中修改为4个新的按钮,即"打开"、"文字"、"缩放"、"画图".

```
<Button
         android:id="@+id/button1"
         android:layout_width="wrap_content"
         android:layout_height="match_parent"
         android:layout_weight="1"
         android:text="打开" />
<Button
         android:id="@+id/button2"
         android:layout_width="wrap_content"
         android:layout_height="match_parent"
         android:layout_weight="1"
         android:text="文字" />
<Button
         android:id="@+id/button3"
         android:layout_width="wrap_content"
         android:layout_height="match_parent"
         android:layout_weight="1"
         android:text="缩放" />
<Button
         android:id="@+id/button4"
         android:layout_width="wrap_content"
         android:layout_height="match_parent"
         android:layout_weight="1"
         android:text="画图" />
```

在上面的自定义变量中添加新的变量,如下:

```
//新增按钮
private Button wordAddBn;              //添加文字
private Button changeImageBn;          //缩放图片
private Button drawImageBn;            //绘制图片
```

```
//图片处理时显示备份
                private Bitmap alteredBitmap;          //图片
private Canvas canvas;                  //画布
private Paint paint;                    //画刷
private RelativeLayout layout;
//标识变量  1-显示图片 2-添加文字 3-缩放图片 4-画图
private int flagOnTouch = 0;
```

然后修改打开图片函数,为其创建一个alteredBitmap图像,后面的添加文字和绘制画图都是在此基础上修改:

```
//打开图片
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
         super.onActivityResult(requestCode, resultCode, data);
            if(resultCode==RESULT_OK && requestCode==IMAGE_CODE) {
                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
                try {
                        //标识变量=1 图片显示
                flagOnTouch = 1;
                    //载入图片尺寸大小没载入图片本身 true
                    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);
                    //inSampleSize表示图片占原图比例 =1表示原图
                    if(heightRatio>1&&widthRatio>1) {
                        if(heightRatio>widthRatio) {
                            bmpFactoryOptions.inSampleSize = heightRatio;                          }
                        else {
                            bmpFactoryOptions.inSampleSize = widthRatio;
                        }
                    }
                //图像真正解码 false
                    bmpFactoryOptions.inJustDecodeBounds = false;
                    bmp = BitmapFactory.decodeStream(getContentResolver().openInputStream(imageFileUri),
null, bmpFactoryOptions);
                    //imageShow.setImageBitmap(bmp);                    //显示文件路径
                    String[] filePathColumn= {MediaStore.Images.Media.DATA};
                    Cursor cursor = getContentResolver().query(imageFileUri, filePathColumn, null, null,
null);
                    cursor.moveToFirst(); //将光标移至开头
                    int columnIndex = cursor.getColumnIndex(filePathColumn[0]); //获得图片索引值
                    path = cursor.getString(columnIndex);                    cursor.close();
                    pathText.setText("path="+path);
                   //加载备份图片
                    alteredBitmap = Bitmap.createBitmap(bmp.getWidth(), bmp
                                        .getHeight(), bmp.getConfig());
                                canvas = new Canvas(alteredBitmap);  //画布
                                paint = new Paint(); //画刷
                                paint.setColor(Color.GREEN);
                                paint.setStrokeWidth(5);
                                paint.setTextSize(30);
                                paint.setTypeface(Typeface.DEFAULT_BOLD);  //无线粗体
                                matrix = new Matrix();
                                canvas.drawBitmap(bmp, matrix, paint);
```

```
                    imageShow.setImageBitmap(alteredBitmap);

                } catch(FileNotFoundException e) {
                    e.printStackTrace();
                }
            }  //end if
}
```

然后在onCreate中添加图片缩放和绘制图片函数,代码如下:

```
//缩放图片 点击按钮"缩放"
changeImageBn = (Button) findViewById(R.id.button3);
changeImageBn.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                flagOnTouch = 3; //缩放
        }
});
//绘制画图 点击按钮"绘制"
drawImageBn = (Button) findViewById(R.id.button4);
drawImageBn.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                flagOnTouch = 4; //绘图
                //画图 图片移动至(0,0) 否则绘图线与手指存在误差
                matrix = new Matrix();
                matrix.postTranslate(0, 0);
                imageShow.setImageMatrix(matrix);
                imageShow.setImageBitmap(alteredBitmap);
        canvas.drawBitmap(bmp, matrix, paint);
        }
 });
```

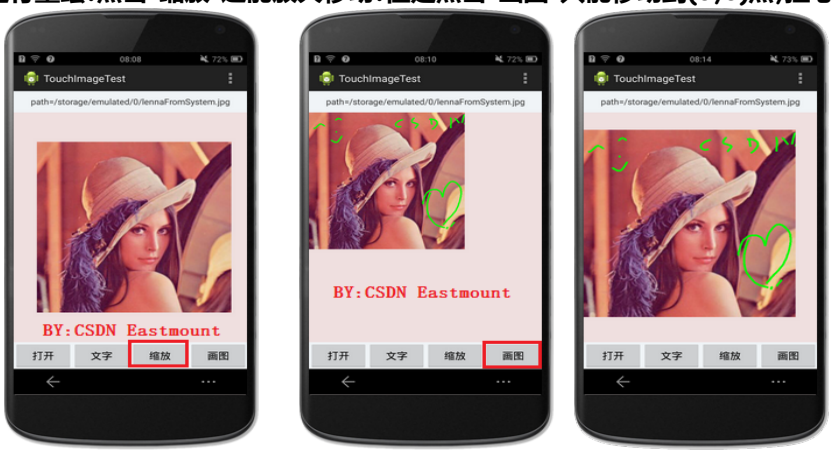
然后修改 imageShow.setOnTouchListener(new OnTouchListener())函数,实现点击不同按钮响应不同的触摸事件:

```
//触屏缩放图片监听 注:XML中修改android:scaleType="matrix"
imageShow.setOnTouchListener(new OnTouchListener() {
        //设置两个点 按下坐标(downx, downy)和抬起坐标(upx, upy)
        float downx = 0;
        float downy = 0;
        float upx = 0;
        float upy = 0;
        //触摸事件
        @Override
        public boolean onTouch(View v, MotionEvent event) {
                ImageView view = (ImageView) v;
                if(flagOnTouch == 1) { //显示图片
                        return true;
                }
                else if(flagOnTouch == 3)  { //图片缩放
                        switch (event.getAction() & MotionEvent.ACTION_MASK) {
                        case MotionEvent.ACTION_DOWN: //手指按下
                                savedMatrix.set(matrix);
                                startPoint.set(event.getX(), event.getY());
                                mode = DRAG;
                                break;
                        case MotionEvent.ACTION_UP:
```

```
                        case MotionEvent.ACTION_POINTER_UP:                                                     mode = NONE;
                                break;
                        case MotionEvent.ACTION_POINTER_DOWN:
                                oldDist = spacing(event); //如果两点距离大于10 多点模式
                                if (oldDist > 10f) {
                                        savedMatrix.set(matrix);
                                        midPoint(middlePoint, event);
                                        mode = ZOOM;
                                }
                                break;
                        case MotionEvent.ACTION_MOVE:
                                if (mode == DRAG) { //拖动
                                        matrix.set(savedMatrix);
                                        matrix.postTranslate(event.getX() - startPoint.x, event.getY() -
startPoint.y);
                                } else if (mode == ZOOM) { //缩放
                                        float newDist = spacing(event);
                                        if (newDist > 10f) {
                                                matrix.set(savedMatrix);
                                                float scale = newDist / oldDist;
                                                matrix.postScale(scale, scale, middlePoint.x, middlePoint.y);
                                        }                                       }
                                break;
                        } //end switch
                        view.setImageMatrix(matrix);
                        return true;
                }
                else if(flagOnTouch == 2) { //图片文字添加
                        return true;
        }
                else if(flagOnTouch == 4) { //绘制图像
                        switch (event.getAction()) {
                                case MotionEvent.ACTION_DOWN:
                                        downx = event.getX();
                                        downy = event.getY();
                                        break;
                                case MotionEvent.ACTION_MOVE:
                                        upx = event.getX();
                                        upy = event.getY();
                                        canvas.drawLine(downx, downy, upx, upy, paint);
                                        imageShow.invalidate();
                                        downx = upx;
                                        downy = upy;
                                        break;
                                case MotionEvent.ACTION_UP:
                                        upx = event.getX();
                                        upy = event.getY();
                                        canvas.drawLine(downx, downy, upx, upy, paint);
                                        imageShow.invalidate();
                                        break;
                                case MotionEvent.ACTION_CANCEL:
                                        break;
                                default:
                                        break;
                                }
                                return true;
                }
        else {
                return false;
        }
        }  //end  onTouch
```

```
        //两点距离
                        private float spacing(MotionEvent event) {
                float x = event.getX(0) - event.getX(1);
                float y = event.getY(0) - event.getY(1);
                return FloatMath.sqrt(x * x + y * y);
        }
        //两点中点
                private void midPoint(PointF point, MotionEvent event) {
                        float x = event.getX(0) + event.getX(1);
                        float y = event.getY(0) + event.getY(1);
                        point.set(x / 2, y / 2);
                }
});
```

此时图像显示的效果如下图所示,点击"打开"按钮只能打开图片,点击"画图"按钮能在原图上绘制,在此点击"画图"可以进行重绘.点击"缩放"还能放大移动.但是点击"画图"只能移动到(0,0)点,担心坐标变换出现不一致现象.

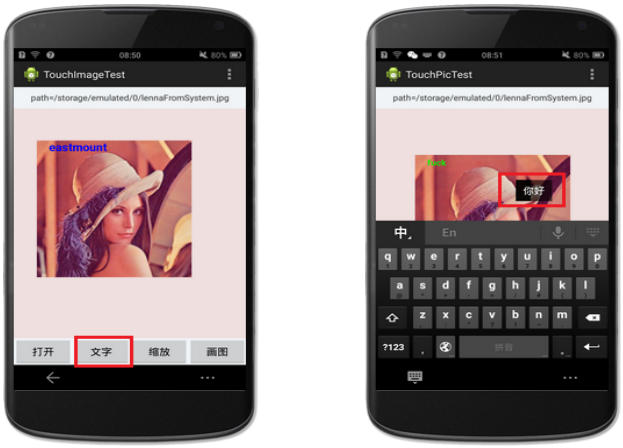


2.添加水印

添加如下函数实现添加水印,主要是通过调用drawText绘制内容:

```
//添加水印文字
wordAddBn = (Button) findViewById(R.id.button2);
layout = (RelativeLayout) findViewById(R.id.Content_Layout);
wordAddBn.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
                flagOnTouch = 2;
                //添加文字
                Bitmap bmpTemp = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(), bmp.getConfig());
        Canvas cv = new Canvas(bmpTemp);
        Paint p = new Paint();
        Typeface font = Typeface.create("宋体", Typeface.BOLD);
        p.setColor(Color.BLUE);
        p.setTypeface(font);
        p.setTextSize(40);
        cv.drawBitmap(bmp, 0, 0, p);
        imageShow.setImageBitmap(bmpTemp);
        cv.drawText("eastmount", 40, 40, p);
        cv.save(Canvas.ALL_SAVE_FLAG);
        cv.restore();
        }
});
```

显示效果如下图所示：



我想实现的效果是点击"文字"按钮后弹出键盘输入文字水印，并且可以拖动文字的效果.但效果不是很好如右图，所以没写出来，只能添加固定的文字.下面的代码是添加EditView代码仅供参考：

```
//键盘添加EditText
private EditText createEditText() {
        EditText et = new EditText(this);
        LayoutParams params = new LayoutParams(LayoutParams.WRAP_CONTENT,
                        LayoutParams.WRAP_CONTENT);
        et.setLayoutParams(params);
        return et;
}
//键盘实现
private void addWigdet(RelativeLayout layout, int x, int y, EditText v) {
        RelativeLayout.LayoutParams params = new RelativeLayout.LayoutParams(
                        LayoutParams.WRAP_CONTENT, LayoutParams.WRAP_CONTENT);
        params.leftMargin = x;
        params.topMargin = y;
        layout.addView(v, params);
        v.setFocusable(true);
        v.setFocusableInTouchMode(true);
        v.requestFocus();
        v.setBackgroundColor(Color.BLACK);
        v.setTextColor(Color.WHITE);
        v.getBackground().setAlpha(160);
        InputMethodManager inputManager = (InputMethodManager)
                        v.getContext().getSystemService(Context.INPUT_METHOD_SERVICE);
        inputManager.showSoftInput(v, 0);
}
```

然后在触摸事件onTouch中添加addWigdet(layout, x, y, createEditText())代码即可实现，我的if(flagOnTouch==2)没有填写任何内容，就是因为预览效果不是很好，读者自己可以去尝试完成~

3.长按保存

上面的代码你可能觉得当添加文字或绘制后，在次点击图片为啥没变化，那时因为我没有保存.我想实现的功能是点击图片长按保存，在刷新下图片即可完成修改.

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使用监听事件setOnTouchListener实现触摸处理图片的基础文章，包括缩放移动图片、添加水印、绘制图片等.初学Android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/8091941

(By:Eastmount 2014-10-28 夜2点 http://blog.csdn.net/eastmount)

参考资料和相关博文:

- Android截图以及加水印Demo By:tangcheng_ok
- Android 自定义View消除锯齿实现图片旋转,添加边框及文字说明
- Android图片查看支持双击放大缩小、多点触摸(边界处理) By:TMajier
- Android浏览图片，点击放大至全屏效果 By:roamer
- 我的Android笔记（十三）Muilti-touch 双指缩放的实现探索 By:barryhappy