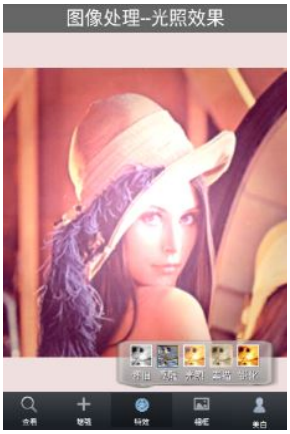
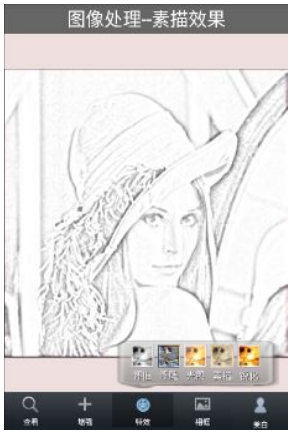
[Android] 图像处理整合之处理ColorMatrix和Intend传递路径显示图像

原创 Eastmount 最后发布于2014-12-03 22:04:59 阅读数 4116 ☆ 收藏 展开

经过几门考试之后,终于有时间整合下自己的Anroid项目"随手拍"的图像处理部分了,主要是结合前面几篇文章讲解的各种android图像处理技术和PopupWindow布局,图像初步整合效果如下.由于该软件目前还未答辩,所以结束后会共享所有的源代码,可能需要一个月后吧,抱歉~





在"随手拍"中点击发布,有添加图片按钮,点击出现"选择本地图片"和"照相截拆选择",显示图片至另一个处理界面,底部五个LinearLayout布局(查看|增强|效果|相框|美白)通过PopupWindow实现弹出窗体,各种处理供选择.

在这篇这次整合中我主要想讲解四个问题:

(PS:我也不知道更好的整合方法,希望有好的android项目整合方法与作者探讨)

1.如何通过Intend实现给其他活动传参,并且传递图像路径string,实现在另外一个窗体显示图像;

2.如何实现把具体图像处理算法定义不同类,处理界面通过Import引入并调用;

3.如何通过SeekBar实现图像的ColorMatrix处理,包括饱和度、图像RGB和亮度;

4.最后简单介绍SeekBar和ColorMatrix的使用方法.

显示效果图如下:

一．Intend传递路径给其他活动显示图像

该方法主要通过Intent实现,通过其putExtra()方法存储要传递的图片路径于Intent中,启动另外一个活动时通过getStringExtra取出数据即可.核心代码:

```
//主活动传递图片path给处理活动
Intent intent = new Intent(this, ProcessActivity.class);
intent.putExtra("path", path);
startActivity(intent);
//处理活动中取数据 注意参数对应putExtra("path", path)
Intent intent = getIntent();
String path = intent.getStringExtra("path");
ShowPhotoByImageView(path); //自定义显示图片函数
```

具体代码操作如下,在activity_main.xml文件中定义一个按钮"button1",再在activity_process.xml中定义图片处理的布局如下:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >
    <RelativeLayout
        android:id="@+id/layout_bottom"
        android:layout_width="fill_parent"
        android:layout_height="110dp"
        android:layout_alignParentBottom="true"
        android:gravity="center">
            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical" >
                <LinearLayout
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:paddingTop="2dp"
                android:orientation="horizontal" >
                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                                android:paddingLeft="20px"
                                android:paddingTop="10px"
                                android:text="饱和" />
                <SeekBar
```

```
                android:id="@+id/seekBarSaturation"
                android:layout_width="fill_parent"
                        android:layout_height="wrap_content"
                        android:max="255"
                                                                    android:progress="127"
                        android:secondaryProgress="75" >
            </SeekBar>
        </LinearLayout>
        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal" >
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                        android:paddingLeft="20px"
                        android:text="色相" />
            <SeekBar
                android:id="@+id/seekBarHue"
                android:layout_width="fill_parent"
                        android:layout_height="wrap_content"
                        android:max="255"
                        android:progress="127"
                        android:secondaryProgress="75" >
            </SeekBar>
        </LinearLayout>
        <LinearLayout
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal" >
            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                        android:paddingLeft="20px"
                        android:text="亮度" />
            <SeekBar
                android:id="@+id/seekBarLum"
                android:layout_width="fill_parent"
                        android:layout_height="wrap_content"
                        android:max="255"
                        android:progress="127"
                        android:secondaryProgress="75" >
            </SeekBar>
        </LinearLayout>
        </LinearLayout>
    </RelativeLayout>
    <!-- 添加layout_above才显示顶部布局 -->
    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_above="@id/layout_bottom"
        android:background="#EFDFDF"
        android:layout_alignParentTop="true"
        android:gravity="center" >
        <ImageView
                android:id="@+id/imageView1"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:layout_gravity="center_horizontal" />
    </RelativeLayout>
</RelativeLayout>
```

然后MainActivity.java文件选择图片并给Intent中装载选择图片的路径,传递给ProcessActivity.java文件处理.

```
public class MainActivity extends Activity {

        //自定义变量
        private Button selectPhoto;        //选择图片
        private final int IMAGE_OPEN = 1;  //打开图片标记

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //选择图片并传递图片路径信息给处理图片活动
        selectPhoto = (Button) findViewById(R.id.button1);
        selectPhoto.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_OPEN);
                }
        });
    }

    //获取图片路径 响应startActivityForResult
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        //打开图片
        if(resultCode==RESULT_OK && requestCode==IMAGE_OPEN) {
                Uri uri = data.getData();
                if (!TextUtils.isEmpty(uri.getAuthority())) {
                        //查询选择图片
                                Cursor cursor = getContentResolver().query(
                                                uri,
                                                new String[] { MediaStore.Images.Media.DATA },
                                                null,
                                                null,
                                                null);
                                //返回 没找到选择图片
                                if (null == cursor) {
                                        return;
                                }
                                //光标移动至开头 获取图片路径
                                cursor.moveToFirst();
                                String path = cursor.getString(cursor
                                                .getColumnIndex(MediaStore.Images.Media.DATA));
                                //Toast.makeText(this, path, Toast.LENGTH_SHORT).show();
                                //向处理活动传递数据
                                Intent intent = new Intent(this, ProcessActivity.class); //主活动->处理活动
                                intent.putExtra("path", path);
                                startActivity(intent);
                        } else {
                                Intent intent = new Intent(this, ProcessActivity.class); //主活动->处理活动
                                intent.putExtra("path", uri.getPath());
                                startActivity(intent);
                        }
        }  //end if 打开图片
    }
}
```

然后是ProcessActivity.java中具体处理,通过自定义函数ShowPhotoByImageView(String path)获取传递的图像路径并显示,代码如下:

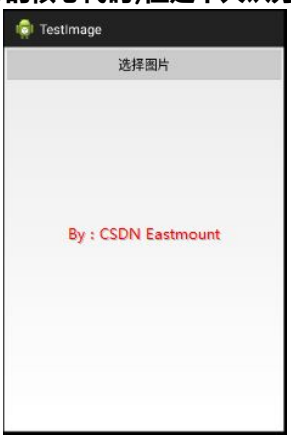
```
public class ProcessActivity extends Activity {

        //自定义变量
        private ImageView imageShow;         //显示图片
        private Bitmap bmp;                  //载入图片
        private SeekBar seekBar1;            //SeekBar 饱和度
        private SeekBar seekBar2;            //SeekBar 色相
        private SeekBar seekBar3;            //SeekBar 亮度

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                 super.onCreate(savedInstanceState);
                 requestWindowFeature(Window.FEATURE_NO_TITLE);
             setContentView(R.layout.activity_process);
             //获取控件
             imageShow = (ImageView) findViewById(R.id.imageView1);
             //载入数据
             Intent intent = getIntent();
             String path = intent.getStringExtra("path"); //对应putExtra("path", path);
             //自定义函数 显示图片
             ShowPhotoByImageView(path);
        }

        /*
         * 函数功能 显示图片
         * 参数 String path 图片路径,源自MainActivity选择传参
         */
        private void ShowPhotoByImageView(String path)
        {
                if (null == path) {
                        Toast.makeText(this, "载入图片失败", Toast.LENGTH_SHORT).show();
                        finish();
                }
                //获取分辨率
                DisplayMetrics dm = new DisplayMetrics();
            getWindowManager().getDefaultDisplay().getMetrics(dm);
            int width = dm.widthPixels;    //屏幕水平分辨率
            int height = dm.heightPixels;  //屏幕垂直分辨率
            try {
                //Load up the image's dimensions not the image itself
                BitmapFactory.Options bmpFactoryOptions = new BitmapFactory.Options();
                bmpFactoryOptions.inJustDecodeBounds = true;
                bmp = BitmapFactory.decodeFile(path,bmpFactoryOptions);
                int heightRatio = (int)Math.ceil(bmpFactoryOptions.outHeight/(float)height);
                int widthRatio = (int)Math.ceil(bmpFactoryOptions.outWidth/(float)width);
                //压缩显示 值为16时很模糊且图片大小没变化
                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.decodeFile(path,bmpFactoryOptions);
```

```
                imageShow.setImageBitmap(bmp); //显示照片
            } catch(Exception e) {
                e.printStackTrace();
            }
        }
    }
```

同时需要注意需要在AndroidManifest.xml中注册活动,否则会报错无法响应;因为需要操作SD卡,所以还需要声明权限.如下所示:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.testimage"
    android:versionCode="1"
    android:versionName="1.0" >
    <!-- 申明权限 操作SD卡 -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" /></span>
    <uses-sdk
        android:minSdkVersion="19"
        android:targetSdkVersion="19" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.testimage.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <!-- 注册新活动 -->
        <activity android:name=".ProcessActivity">
        </activity></span>
    </application>
</manifest>
```

此时运行效果如下图所示,能显示一张图片,第一部分主要是讲述通过Intent传递值并显示图片,其主要内容就是开头部分的核心代码,但是个人认为对初学者来说我认为还是非常有用的.



二. Import导入其他java文件调用其处理函数

在使用android时,我发现它引用其它类函数非常方便,具体方法如下:

```
//自定义类 实现饱和度、色相、亮度图像增强处理
public class IncreaseProcessImage
{
        private Bitmap mBitmap;
        //构造函数
        public IncreaseProcessImage(Bitmap bmp)
        {
                        mBitmap = bmp;
        }
        //图像增强
        //flag=0表示是否改变饱和度 flag=1表示是否改变色相 flag=2表示是否改变亮度
        public Bitmap IncreaseProcessing(Bitmap bmp, int flag)
        {
                ...
                return bitmap;
        }

}
```

在处理类中调用其它类的方法代码如下:

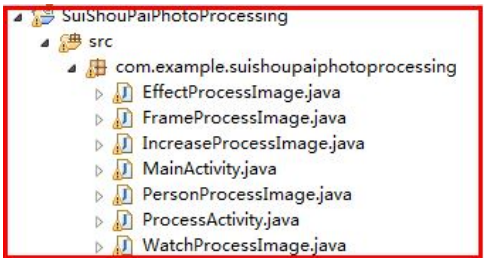
```
//引入图像增强处理类
import com.example.testimage.IncreaseProcessImage;

public class ProcessActivity extends Acticity {
        ...
        //调用IncreaseProcessImage类中处理方法
        IncreaseProcessImage increaseProcess = null;
        increaseProcess = new IncreaseProcessImage(bmp);
        Bitmap mbmp = increaseProcess.IncreaseProcessing(bmp,flag);
        imageShow.setImageBitmap(mbmp);
        ...
}
```

其实我一直都不太会整合项目,不论是C#、C++还是Java的项目,也不知道如何去提高.而且这部分文章也有点少,很多都是技术文章和提高代码质量的,我的"随手拍"基本形式就如下:

MainAcitvity中定义具体的打开图片操作和拍照;

ProcessActivity中是具体的图像处理操作,其中分为5个部分分别调用其他的EffectProcessImage.java(效果处理)、Frame(框架处理)、Increase(增强处理)、Watch(缩放查看)、Person(美白处理).每个java对应的是相应的处理函数.

希望以后能学到更好的整合方法,同时肯定需要以后的工作经验来完善~



三. ColorMatrix图像颜色处理及SeekBar控件详解

在图像增强部分主要通过ColorMatrix图像颜色处理及SeekBar控件实现.直接上代码吧!此部分主要参照那本图像处理的书.

IncreaseProcessImage.java类

```
//自定义类实现图像增强效果处理 包括:饱和度、色相、亮度
public class IncreaseProcessImage
{
        private Bitmap mBitmap;
        //构造函数
        public IncreaseProcessImage(Bitmap bmp)
        {
                mBitmap = bmp;
        }
        //自定义变量
        private ColorMatrix mSaturationMatrix;     //饱和度
        private ColorMatrix mHueMatrix;            //色相
        private ColorMatrix mLumMatrix;            //亮度
        private ColorMatrix mAllMatrix;
        private float mSaturationValue = 0F;
        private float mHueValue = 0F;
        private float mLumValue = 0F;
        //SeekBar中间值127 [0-255]
        private static final int MIDDLE_VALUE = 127;
        private static final int MAX_VALUE =255;
        /*
         * 设置饱和度值
         */
        public void setSaturation(int value) {
                mSaturationValue = value * 1.0F / MIDDLE_VALUE;
        }
        /*
         * 设置色相值
         */
        public void SetHue(int value) {
                mHueValue = (value - MIDDLE_VALUE) * 1.0F / MIDDLE_VALUE  * 180;
        }
        /*
         * 设置亮度值
         */
        public void SetLum(int value) {
                mLumValue = value * 1.0F / MIDDLE_VALUE;
        }
        /*
         * 图像增强
         * 饱和度处理 色相处理 亮度处理
         * flag=0表示是否改变饱和度 flag=1表示是否改变色相 flag=2表示是否改变亮度
         */
        public Bitmap IncreaseProcessing(Bitmap bmp, int flag)
        {
                //创建一个相同尺寸可变的位图区,用于绘制新图
                Bitmap bitmap = Bitmap.createBitmap(bmp.getWidth(), bmp.getHeight(),
                                Bitmap.Config.ARGB_8888);
                Canvas canvas = new Canvas(bitmap);
                Paint paint = new Paint();
                paint.setAntiAlias(true);       //给Canvas加上抗锯齿标志,边缘平滑处理
                if(mAllMatrix == null) {
                        mAllMatrix = new ColorMatrix();
                }
                if(mSaturationMatrix == null) {
```

```
                mSaturationMatrix = new ColorMatrix();                }
            if(mHueMatrix == null) {
                mHueMatrix = new ColorMatrix();
            }
            if(mLumMatrix == null) {
                mLumMatrix = new ColorMatrix();
            }
            //图像处理
            if(flag==0) { //饱和度
                //饱和度值最小可设为0,此时表示灰度图 为1表示饱和度不变,设置大于1就是过饱和
                mSaturationMatrix.reset(); //设为默认值
                mSaturationMatrix.setSaturation(mSaturationValue);
            }
            else if(flag==1) { //色相
                //hueColor是色轮旋转角度,正值表示顺时针旋转,负值表示逆时针旋转
                mHueMatrix.reset();
                mHueMatrix.setRotate(0, mHueValue); //红色区在色轮上旋转的角度
                mHueMatrix.setRotate(1, mHueValue); //绿色区在色轮上旋转的角度
                mHueMatrix.setRotate(2, mHueValue); //蓝色取在色轮上旋转的角
度
            }                else if(flag==2) { //亮度
                mLumMatrix.reset();
                //红绿蓝三色按相同比例,最后参数1表示透明度不变
                mLumMatrix.setScale(mLumValue, mLumValue, mLumValue, 1);
            }
            //设置AllMatrix
            mAllMatrix.reset();
            mAllMatrix.postConcat(mHueMatrix); //效果叠加
            mAllMatrix.postConcat(mSaturationMatrix);
            mAllMatrix.postConcat(mLumMatrix);
            //设置颜色变换效果
            paint.setColorFilter(new ColorMatrixColorFilter(mAllMatrix));
            canvas.drawBitmap(bmp, 0, 0, paint); //颜色变化后输出到新创建位图区
            return bitmap;
        }
        /*
         * End
         */
}
```

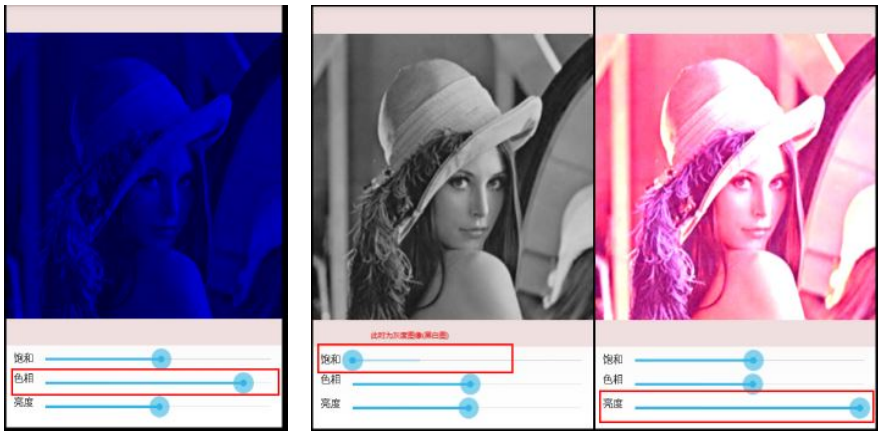
然后修改ProcessActivity.java处理活动,同时需要为活动ProcessActivity添加接口OnSeekBarChangeListener,修改后代码如下:

```
//引入图像增强处理类
import com.example.testimage.IncreaseProcessImage;

public class ProcessActivity extends Activity implements OnSeekBarChangeListener {

        //自定义变量
        private ImageView imageShow;              //显示图片
        private Bitmap bmp;                       //载入图片
        private SeekBar seekBar1;                 //SeekBar 饱和度
        private SeekBar seekBar2;                 //SeekBar 色相
        private SeekBar seekBar3;                 //SeekBar 亮度
        IncreaseProcessImage increaseProcess = null;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
                 super.onCreate(savedInstanceState);
```

```
        requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_process);
                                                        //获取控件
    imageShow = (ImageView) findViewById(R.id.imageView1);
    seekBar1 = (SeekBar) findViewById(R.id.seekBarSaturation);  //饱和度
 seekBar2 = (SeekBar) findViewById(R.id.seekBarHue);            //色相
 seekBar3 = (SeekBar) findViewById(R.id.seekBarLum);            //亮度
    //载入数据
    Intent intent = getIntent();
    String path = intent.getStringExtra("path"); //对应putExtra("path", path);
    //自定义函数 显示图片
    ShowPhotoByImageView(path);
    //自定义函数 设置SeekBar监听事件
    SetClickTouchListener();
}

/*
 * 函数功能 显示图片
 * 参数 String path 图片路径,源自MainActivity选择传参
 */
private void ShowPhotoByImageView(String path)
{
        if (null == path) {
                Toast.makeText(this, "载入图片失败", Toast.LENGTH_SHORT).show();
                finish();
        }
        //获取分辨率
        DisplayMetrics dm = new DisplayMetrics();
    getWindowManager().getDefaultDisplay().getMetrics(dm);
    int width = dm.widthPixels;    //屏幕水平分辨率
    int height = dm.heightPixels;  //屏幕垂直分辨率
    try {
        //Load up the image's dimensions not the image itself
        BitmapFactory.Options bmpFactoryOptions = new BitmapFactory.Options();
        bmpFactoryOptions.inJustDecodeBounds = true;
        bmp = BitmapFactory.decodeFile(path,bmpFactoryOptions);
        int heightRatio = (int)Math.ceil(bmpFactoryOptions.outHeight/(float)height);
        int widthRatio = (int)Math.ceil(bmpFactoryOptions.outWidth/(float)width);
        //压缩显示 值为16时很模糊且图片大小没变化
        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.decodeFile(path,bmpFactoryOptions);
        imageShow.setImageBitmap(bmp); //显示照片
    } catch(Exception e) {
    e.printStackTrace();
}
}
 //设置SeekBar监听事件
private void SetClickTouchListener()
{
        /*
      * 设置Seekbar变化监听事件
      * 注意:此时为活动ProcessActivity添加接口
```

```
         * ProcessActivity extends Activity implements OnSeekBarChangeListener                */
        seekBar1.setOnSeekBarChangeListener(this);
        seekBar2.setOnSeekBarChangeListener(this);
        seekBar3.setOnSeekBarChangeListener(this);
        increaseProcess = new IncreaseProcessImage(bmp);
    }
    //拖动值监听
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromTouch)
    {
        int flag = 0;
        switch(seekBar.getId()) {
        case R.id.seekBarSaturation: //饱和度
            flag = 0;
            increaseProcess.setSaturation(progress);
            break;
        case R.id.seekBarHue: //色相
            flag = 1;
            increaseProcess.SetHue(progress);
            break;
        case R.id.seekBarLum: //亮度
            flag = 2;
            increaseProcess.SetLum(progress);
            break;
        }
        imageShow.setImageBitmap(increaseProcess.IncreaseProcessing(bmp,flag));
    }
    //SeekBar 开始拖动 否则ProcessActivity报错
    public void onStartTrackingTouch(SeekBar seekBar)
    {

    }
    //SeekBar 停止拖动
    public void onStopTrackingTouch(SeekBar seekBar)
    {

    }
}
```

此时运行效果如下：



四. 进度条SeekBar和ColorMatrix使用介绍

SeekBar(拖动条)类似于进度条，如声音视频进度条.拖动可以被用户控制，需要对其进行事件监听，所以需要实现SeekBar.OnSeekBarChangeListener接口.在SeekBar中需要监听3个事件：

```
//监听数值改变 通常使用它监听值变化
public void onProgressChanged(SeekBar seekBar,int progress, boolean fromTouch);
                                                                    //监听开始拖动
public void onStartTrackingTouch(SeekBar seekBar);
//监听停止拖动
public void onStopTrackingTouch(SeekBar seekBar);
```

其中在XML中可以设置其SeekBar常用属性如下:

```
android:max="255"               //设置拖动条最大值
android:progress="100"          //设置当前进度值
android:secondaryProgress="75" //设置第二进度,通常视频缓冲时会见到该效果
android:progressDrawable="@drawable/" //设置进图条图样
android:thumb="@drawable/"             //设置滑块图样 可定义xml设置其变换
```

而ColorMatrix主要介绍下面官方文档常用几个函数(android.graphics.ColorMatrix)

1.void postConcat(ColorMatrix postmatrix)

Concat this colormatrix with the specified postmatrix.

将当前颜色矩阵连接到postmatrix矩阵后边,相当于setConcat(postmatrix, this).

2.void preConcat(ColorMatrix prematrix)

Concat this colormatrix with the specified prematrix.

将prematrix连接到当前颜色矩阵后边,相当于调用setConcat(this, prematrix).

3.public void reset ()

Set this colormatrix to identity: [1 0 0 0 0 - red vector 0 1 0 0 0 - green vector 0 0 1 0 0 - blue vector 0 0 0 1 0] - alpha vector

设置当前颜色矩阵的数组变成标准数组,默认设置.

4.public void setSaturation (float sat)

Set the matrix to affect the saturation of colors.

设置颜色饱和度,0表示灰度图(俗称的黑白图),1表示饱和度不变,大于1表示过饱和显示.

5.public void setScale (float rScale,float gScale,float bScale,float aScale)

Set this colormatrix to scale by the specified values.

改变颜色矩阵数据,变换时缩小颜色的ARGB值,最后一个参数1表示透明度不变.

6.public void setRotate (int axis, float degrees)

设置图片中的颜色可以选择一条轴线axis来旋转degrees度,当axis=0表示RED为轴线,当axis=1时表示GREEN为轴线,当axis=2时表示BLUE为轴线.正值表示顺时针旋转,负值表示逆时针旋转.

推荐自定义SeekBar各种美观样式博客:

自定义漂亮的Android SeekBar样式 By:w8320273

自定义SeekBar By:imdxt1986

最后希望文章对大家有所帮助吧~如果有什么错误或不足之处,请海涵.文章对刚学习android处理图像的同学还是有些帮助的,虽然内容比较基础,但主要讲述了我整合过程中遇到的几个问题.

下载地址:http://download.csdn.net/detail/eastmount/8222493

(By:Eastmount 2014-12-3 晚上10点 http://blog.csdn.net/eastmount/)