王亮 Java/android学习

目录视图 摘要视图 RSS 订阅

个人资料

xjwangliang

访问： 54285次

积分： 961

等级： BLOG 3

排名： 千里之外

原创： 37篇 转载： 31篇

译文： 1篇 评论： 5条

文章搜索

文章分类

Java	(11)
android	(37)
python	(4)
Lua	(2)
shell	(1)
linux-c	(1)
rxjava	(3)
Git	(2)

文章存档

2016年04月	(5)
2016年03月	(8)
2015年06月	(2)
2015年03月	(1)
2014年08月	(3)
2014年06月	(1)
2014年05月	(2)
2014年03月	(10)
2014年01月	(3)
2013年11月	(4)

android **JNI**处理**图片**的例子

标签： android **jni** image processing structure ibm

2011-12-13 09:52 9055人阅读 评论(3) 收藏 举报

分类： android（36）

版权声明：本文为博主原创文章，未经博主允许不得转载。

//下面是主程序代码

```
import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.Bitmap.Config;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;

public class PhotoShop extends Activity {
        private String tag = "IBMPhotoPhun";
        private Bitmap bitmapOrig = null;
        private Bitmap bitmapGray = null;
        private Bitmap bitmapWip = null;
        private ImageView ivDisplay = null;
        // NDK STUFF
        static {
                try{
                System.loadLibrary("haha");}
                catch(Exception e) {
                        System.out.println("dd");
                }
        }
        public native void convertToGray(Bitmap bitmapIn, Bitmap bitmapOut);
```

2013年09月 (1)
2013年08月 (3)
2012年02月 (1)
2012年01月 (2)
2011年12月 (12)
2011年07月 (3)
2011年06月 (8)

阅读排行

android **JNI**处理图片的	(9054)
android_Lame转换音频	(3897)
VideoView笔记	(3858)
AutoCompleteTextView	(2399)
Genymotion 2.0之后安装	(2207)
使用TextView/EditText应	(1798)
android反射获取资源	(1511)
IDEA+maven使用最新版	(1463)
Fragment保存状态	(1330)
Bitmap和Canvas的平移	(882)

评论排行

android **JNI**处理图片的	(3)
android_Lame转换音频	(1)
VideoView笔记	(1)
关于信息隐藏等等	(0)
关于序列化和反序列化的	(0)
文章标题	(0)
多线程笔记	(0)
类的加载笔记	(0)
关于文件File的笔记	(0)
内部类小结	(0)

推荐文章

* 致JavaScript也将征服的物联网世界
* 从苏宁电器到卡巴斯基：难忘的三年硕士时光
* 作为一名基层管理者如何利用情商管理自己和团队（一）
* Android CircleImageView圆形ImageView
* 高质量代码的命名法则

最新评论

android **JNI**处理图片的例子
nos19880918: @maggiccrystal_3:https://www.ibm

android_Lame转换音频格式
mmorss: 这个不对

VideoView笔记
hhd904928: 顶一个，我也看了下源码，你总结的非常好！学习了

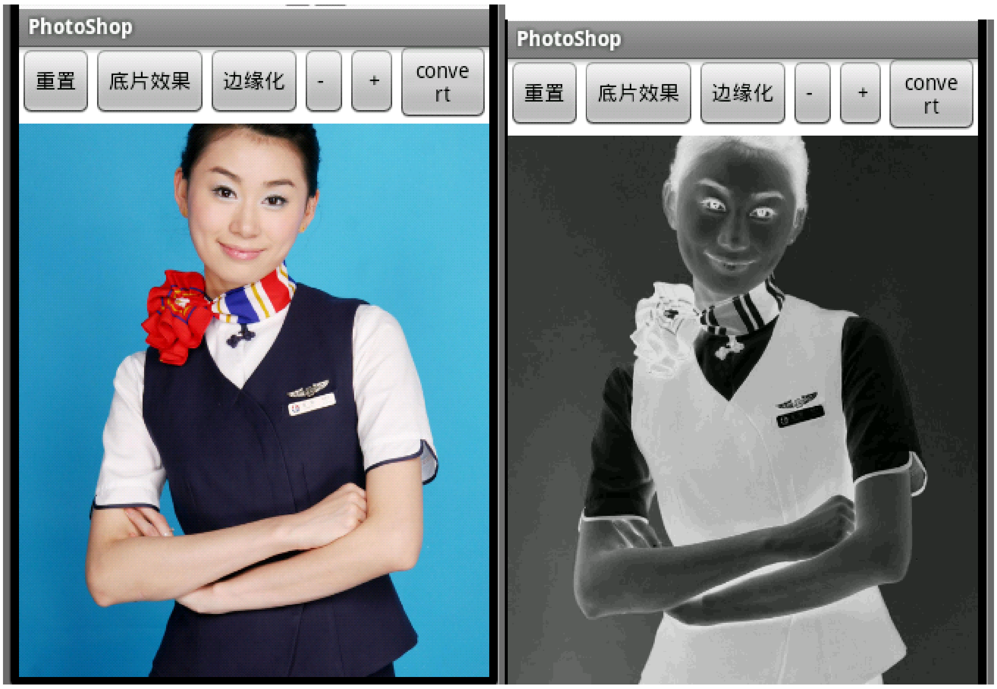
android **JNI**处理图片的例子
oJiangJiaZhi: 使用这个获取的bitmap信息保存成jpg严重失真，请问遇到过吗？

android **JNI**处理图片的例子
maggiccrystal_3: 能否附上源码？谢谢楼主

```
public native void changeBrightness(int direction, Bitmap bitmap);

public native void findEdges(Bitmap bitmapIn, Bitmap bitmapOut);

// END NDK STUFF
/** Called when the activity is first created. */
@Override
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Log.i(tag, "before image stuff");
        ivDisplay = (ImageView) findViewById(R.id.ivDisplay);
        // load bitmap from resources
        BitmapFactory.Options options = new BitmapFactory.Options();
        // Make sure it is 24 bit color as our image processing algorithm
        // expects this format
        options.inPreferredConfig = Config.ARGB_8888;
        bitmapOrig = BitmapFactory.decodeResource(this.getResources(),
                        R.drawable.sample, options);
        if (bitmapOrig != null)
                ivDisplay.setImageBitmap(bitmapOrig);
}

public void onResetImage(View v) {
        Log.i(tag, "onResetImage");
        ivDisplay.setImageBitmap(bitmapOrig);
}
// 边缘化
public void onFindEdges(View v) {
        Log.i(tag, "onFindEdges");
        // make sure our target bitmaps are happy
        bitmapGray = Bitmap.createBitmap(bitmapOrig.getWidth(),
                        bitmapOrig.getHeight(), Config.ALPHA_8);
        bitmapWip = Bitmap.createBitmap(bitmapOrig.getWidth(),
                        bitmapOrig.getHeight(), Config.ALPHA_8);
        // before finding edges, we need to convert this image to gray
        convertToGray(bitmapOrig, bitmapGray);
        // find edges in the image
        findEdges(bitmapGray, bitmapWip);
        ivDisplay.setImageBitmap(bitmapWip);
}

public void onConvertToGray(View v) {
        Log.i(tag, "onConvertToGray");
        //创建一个和原图宽和高相等的灰色（一个像素是一个字节）图片，内容不确定或者为空
(bitmapWip != null)，就是说造好了格子只等填充数据
        bitmapWip = Bitmap.createBitmap(bitmapOrig.getWidth(),
                        bitmapOrig.getHeight(), Config.ALPHA_8);
        /**
         * java中图片的config（相当于格式）
         *  ALPHA_8     (2),
        RGB_565     (4),
        ARGB_4444   (5),
        ARGB_8888   (6);
```

```
                与android不同：
                ANDROID_BITMAP_FORMAT_RGBA_8888:1
                        ANDROID_BITMAP_FORMAT_A_8:8
                 */
                convertToGray(bitmapOrig, bitmapWip);
                ivDisplay.setImageBitmap(bitmapWip);
        }
        //先要获得bitmapWip才能处理，否则setImageBitmap(byll)虽然不会报错，但是空白的
        public void onDimmer(View v) {
                Log.i(tag, "onDimmer");
                changeBrightness(2, bitmapWip);
                ivDisplay.setImageBitmap(bitmapWip);
        }
        //先要获得bitmapWip才能处理，否则setImageBitmap(byll)虽然不会报错，但是空白的
        public void onBrighter(View v) {
                Log.i(tag, "onBrighter");
                changeBrightness(1, bitmapWip);
                ivDisplay.setImageBitmap(bitmapWip);
        }
}


//下面是JNI层的代码

//=====================================================

include <jni.h>
#include <android/log.h>
#include <android/bitmap.h>//【导入】
#define LOG_TAG "libibmphotophun"
#define LOGI(...) __android_log_print(ANDROID_LOG_INFO,LOG_TAG,__VA_ARGS__)
#define LOGE(...) __android_log_print(ANDROID_LOG_ERROR,LOG_TAG,__VA_ARGS__)
typedef struct//彩色图片数据
{
        uint8_t alpha;
        uint8_t red;
        uint8_t green;
        uint8_t blue;
} argb;
/*
convertToGray
Pixel operation
bitmapcolor：24位（实际是32）彩色图片；格式：ANDROID_BITMAP_FORMAT_RGBA_8888：
bitmapgray灰色图片（8位）；格式：ANDROID_BITMAP_FORMAT_A_8
根据彩色图片填充灰色图片数据
*/
JNIEXPORT void JNICALL Java_com_msi_ibm_ndk_IBMPhotoPhun_convertToGray(JNIEnv
* env, jobject obj, jobject bitmapcolor,jobject bitmapgray)
{
        AndroidBitmapInfo       infocolor;//AndroidBitmapInfo图片信息
        void*                           pixelscolor;//图片地址
        AndroidBitmapInfo       infogray;
        void*                           pixelsgray;
        int                                     ret;
        int                                     y;
        int                                     x;
        LOGI("convertToGray");

        //AndroidBitmap_getInfo(env, bitmapcolor, &infocolor)得到图片信息
        //AndroidBitmap_lockPixels(env, bitmapcolor, &pixelscolor)得到图片地址

        if ((ret = AndroidBitmap_getInfo(env, bitmapcolor, &infocolor)) < 0) {//【AndroidBitmap_getInfo】没有图片信息（宽高等等）
                LOGE("AndroidBitmap_getInfo() failed ! error=%d", ret);
                return;
        }
        if ((ret = AndroidBitmap_getInfo(env, bitmapgray, &infogray)) < 0) {
                LOGE("AndroidBitmap_getInfo() failed ! error=%d", ret);
                return;
        }
        LOGI("color image :: width is %d; height is %d; stride is %d; format is %d;flags is%d",
        infocolor.width,infocolor.height,infocolor.stride,infocolor.format,infocolor.flags);//【获得图片的宽和高、格式】

        if (infocolor.format != ANDROID_BITMAP_FORMAT_RGBA_8888) {//不是24位图片
                LOGE("Bitmap format is not RGBA_8888 !");
                return;
        }
        LOGI("gray image :: width is %d; height is %d; stride is %d; format is %d;flags is
        %d",infogray.width,infogray.height,infogray.stride,infogray.format,infogray.flags);

        if (infogray.format != ANDROID_BITMAP_FORMAT_A_8) {//不是8位图片
                LOGE("Bitmap format is not A_8 !");
                return;
        }
        if ((ret = AndroidBitmap_lockPixels(env, bitmapcolor, &pixelscolor)) < 0) {//【锁定AndroidBitmap_lockPixels】
                //锁定之后，pixelscolor指向图片的首地址
                LOGE("AndroidBitmap_lockPixels() failed ! error=%d", ret);
        }
        if ((ret = AndroidBitmap_lockPixels(env, bitmapgray, &pixelsgray)) < 0) {
                //锁定之后，pixelscolor指向图片的首地址
                LOGE("AndroidBitmap_lockPixels() failed ! error=%d", ret);
        }
        // modify pixels with image processing algorithm
        for (y=0;y<infocolor.height;y++) {//infocolor.height获得图片的高度
                //这是首次使用argb
                argb * line = (argb *) pixelscolor;//每一行的首地址（刚开始是图片的首地址，即第一行的首地址，下面会换行）

                uint8_t * grayline = (uint8_t *) pixelsgray;//每一行的首地址（数据类型为一个无符号字节）

                for (x=0;x<infocolor.width;x++) {
                        grayline[x] = 0.3 * line[x].red + 0.59 * line[x].green + 0.11*line[x].blue;
                }
                pixelscolor = (char *)pixelscolor + infocolor.stride;   //换行：每行的首地址 + 每行的跨度 = 下一行的首地址
                pixelsgray = (char *) pixelsgray + infogray.stride;             //换行：每行的首地址 + 每行的跨度 = 下一行的首地址
        }
        LOGI("unlocking pixels");
        AndroidBitmap_unlockPixels(env, bitmapcolor);
        AndroidBitmap_unlockPixels(env, bitmapgray);
}
/**
The AndroidBitmapInfo structure, defined in bitmap.h, is helpful for
learning about a Bitmap object.

The AndroidBitmap_getInfo function, found in the jnigraphics library,
obtains information about a specific Bitmap object.

a char represents a signed 8-bit value, so a char pointer (char *) allows you to reference an 8-bit
value and perform operations through that pointer. The image data is represented as uint8_t, which
means an unsigned 8-bit value, where each byte holds a value ranging from 0 to 255. A collection of
three 8-bit unsigned values represents a pixel of image data for a 24-bit image.
图象数据使用unit8_t(8位无符号，0-255)，24-bit的图象就是3个8比特无符号数。

Working through an image involves working on the individual rows of data and
moving across the columns. The Bitmap structure contains a member known as the
stride. The stride represents the width, in bytes, of a row of image data. For
example, a 24-bit color plus alpha channel image has 32 bits, or 4 bytes, per pixel.
So an image with a width of 320 pixels has a stride of 320*4 or 1,280 bytes. An 8-bit
grayscale image has 8 bits, or 1 byte, per pixel. A grayscale bitmap with a width of
320 pixels has a stride of 320*1 or simply 320 bytes. With this information in mind,
let's look at the image processing algorithm for converting a color image to a
grayscale image:
Bitmap结构有一个属性stride（跨度）：表示一行数据的长度。例如，24比特的图象每个像素有包含32位数据
（红绿蓝三种颜色，再加上alpha属性），所以宽度为320像素的图片的跨度（stride）为320*4字节。而rayscale
image每个像素只有8比特，就是一个字节，如果320像素，那么stride为320字节。

下面是原作者解释：
1. When the image data is "locked," the base address of the image data is
referenced by a pointer named pixelscolor for the input color image
and pixelsgray for the output grayscale image.

2. Two for-next loops allow you to iterate over the entire image.

        1. First, you iterate over the height of the image, one pass per "row."
        Use the infocolor.height value to get the count of the rows.

        2. On each pass through the rows a pointer is set up to the memory
        location corresponding to the first "column" of image data for the
        row.

        3. As you iterate over the columns for a particular row, you convert
        each pixel of color data to a single value representing the
        grayscale value.

        4. When the complete row is converted you need to advance the
        pointers to the next row. This is done by jumping forward in
        memory by the stride value.

*/
```

```
/*
changeBrightness
Pixel Operation
改变亮度：direction=1表示增加亮度，direction=2减少亮度
bitmap只接受一个灰色图片（一个像素一个字节，格式ANDROID_BITMAP_FORMAT_A_8）
将灰色图片的像素值改变
*/
JNIEXPORT void JNICALL Java_com_msi_ibm_ndk_IBMPhotoPhun_changeBrightness(JNIEnv
* env, jobject obj, int direction,jobject bitmap)
{
        AndroidBitmapInfo       infogray;
        void*                           pixelsgray;
        int                                     ret;
        int                                     y;
        int                                     x;
        uint8_t                 save;//好像没有用到

        //=======================
        if ((ret = AndroidBitmap_getInfo(env, bitmap, &infogray)) < 0) {
                LOGE("AndroidBitmap_getInfo() failed ! error=%d", ret);
                return;
        }

        LOGI("gray image :: width is %d; height is %d; stride is %d; format is %d;flags is
        %d",infogray.width,infogray.height,infogray.stride,infogray.format,infogray.flags);

        if (infogray.format != ANDROID_BITMAP_FORMAT_A_8) {
                LOGE("Bitmap format is not A_8 !");
                return;
        }
        if ((ret = AndroidBitmap_lockPixels(env, bitmap, &pixelsgray)) < 0) {
                LOGE("AndroidBitmap_lockPixels() failed ! error=%d", ret);
        }
        // modify pixels with image processing algorithm
        LOGI("time to modify pixels....");

        for (y=0;y<infogray.height;y++) {//每个像素
                uint8_t * grayline = (uint8_t *) pixelsgray;
                int v;
                for (x=0;x<infogray.width;x++) {
                        v = (int) grayline[x];
                        if (direction == 1)//与上面不同，灰色图片每个像素是一个字节，所以只需要改变一
个字节
                            v -=5;
                        else
                            v += 5;

                        if (v >= 255) {
                                grayline[x] = 255;
                        } else if (v <= 0) {
                                grayline[x] = 0;
                        } else {
                                grayline[x] = (uint8_t) v;
                        }
                }
                pixelsgray = (char *) pixelsgray + infogray.stride;
        }
        AndroidBitmap_unlockPixels(env, bitmap);
}

/*



/*
findEdges
Matrix operation
利用bitmapgray填充bitmapedges
*/
JNIEXPORT void JNICALL Java_com_msi_ibm_ndk_IBMPhotoPhun_findEdges(JNIEnv
* env, jobject obj, jobject bitmapgray,jobject bitmapedges)
{
        AndroidBitmapInfo       infogray;
        void*                               pixelsgray;
        AndroidBitmapInfo       infoedges;
        void*                               pixelsedge;
        int                                     ret;
        int                                     y;
        int                                     x;
        int                                     sumX,sumY,sum;
        int                                     i,j;
        int                                     Gx[3][3];
        int                                     Gy[3][3];
        uint8_t *graydata;
        uint8_t *edgedata;



        LOGI("findEdges running");

        /**
                [  -1    0    1  ]
        Gx:     [  -2    0    2   ]
                [  -1    0    1  ]

                [   1    2    1  ]
        Gy:     [   0    0    0   ]
                [  -1   -2   -1  ]
        */
        Gx[0][0] = -1;Gx[0][1] = 0;Gx[0][2] = 1;
        Gx[1][0] = -2;Gx[1][1] = 0;Gx[1][2] = 2;
        Gx[2][0] = -1;Gx[2][1] = 0;Gx[2][2] = 1;
        Gy[0][0] = 1;Gy[0][1] = 2;Gy[0][2] = 1;
        Gy[1][0] = 0;Gy[1][1] = 0;Gy[1][2] = 0;
        Gy[2][0] = -1;Gy[2][1] = -2;Gy[2][2] = -1;



        if ((ret = AndroidBitmap_getInfo(env, bitmapgray, &infogray)) < 0) {
                LOGE("AndroidBitmap_getInfo() failed ! error=%d", ret);
                return;
        }
        if ((ret = AndroidBitmap_getInfo(env, bitmapedges, &infoedges)) < 0) {
                LOGE("AndroidBitmap_getInfo() failed ! error=%d", ret);
                return;
        }
        LOGI("gray image :: width is %d; height is %d; stride is %d; format is %d;flags is
        %d",infogray.width,infogray.height,infogray.stride,infogray.format,infogray.flags);
        if (infogray.format != ANDROID_BITMAP_FORMAT_A_8) {
                LOGE("Bitmap format is not A_8 !");
                return;
        }
        LOGI("color image :: width is %d; height is %d; stride is %d; format is %d;flags is
        %d",infoedges.width,infoedges.height,infoedges.stride,infoedges.format,infoedges.flags);
        if (infoedges.format != ANDROID_BITMAP_FORMAT_A_8) {
                LOGE("Bitmap format is not A_8 !");
                return;
        }
        if ((ret = AndroidBitmap_lockPixels(env, bitmapgray, &pixelsgray)) < 0) {
                LOGE("AndroidBitmap_lockPixels() failed ! error=%d", ret);
        }
        if ((ret = AndroidBitmap_lockPixels(env, bitmapedges, &pixelsedge)) < 0) {
                LOGE("AndroidBitmap_lockPixels() failed ! error=%d", ret);
        }
        // modify pixels with image processing algorithm
        LOGI("time to modify pixels....");

        graydata = (uint8_t *) pixelsgray;
        edgedata = (uint8_t *) pixelsedge;

        for (y=0;y<=infogray.height - 1;y++) {
                for (x=0;x<infogray.width -1;x++) {
                sumX = 0;
                sumY = 0;
                // check boundaries
                if (y==0 || y == infogray.height-1) {
                        sum = 0;
                } else if (x == 0 || x == infogray.width -1) {
                        sum = 0;
                } else {
                // calc X gradient
                        for (i=-1;i<=1;i++) {
                                for (j=-1;j<=1;j++) {
                                        sumX += (int) ( (*(graydata + x + i + (y + j)*
infogray.stride)) * Gx[i+1][j+1]);
                                }
                        }
                        // calc Y gradient
                        for (i=-1;i<=1;i++) {
                                for (j=-1;j<=1;j++) {
                                        sumY += (int) ( (*(graydata + x + i + (y + j)*
infogray.stride)) * Gy[i+1][j+1]);
                                }
                        }
                        sum = abs(sumX) + abs(sumY);
                }
                if (sum>255) sum = 255;
                if (sum<0) sum = 0;
                *(edgedata + x + y*infogray.width) = 255 - (uint8_t) sum;//给边界图片bitmapedges填充数据
                }
        }
        AndroidBitmap_unlockPixels(env, bitmapgray);
        AndroidBitmap_unlockPixels(env, bitmapedges);

}//结束
```



顶 0　踩 0

上一篇 SQLite的工具类 ---- 通过反射把Cursor封装到VO对象

下一篇 android_Lame转换音频格式

猜你在找

查看评论

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场