

目前主流app都具有上传头像啊，上传图片的功能，看起来好简单的需求，但是其实这里面有一点点不同的地方。先说一下我的思路，因为开发周期的问题，并没有打算自定义相机与图片查询工具，打算采用系统相机和图片查看工具，最开始我打算调用系统的剪裁并且取得的效果还是不错的，因为我最开始做的是系统头像上传的这个功能，后来我采用同样的方法做了上传商品图片的功能，但是这个时候就暴露了之前的隐患。

我先说一下，安卓系统的默认的机制，Intent触发Camera程序，拍好照片后，将会返回数据，但是考虑到内存问题，Camera不会将全尺寸的图像返回给调用的Activity，一般情况下，有可能返回的是缩略图，比如120*160px。这看起来就像是一个缩略图一样了，这样的效果是让我们非常不满意的，因为我们想要上传的明明是一个高清的图片，但是这样我们上传的竟然是一个特别模糊的图片啦，所以我们不能采用这种办法了。

我们的思路：调用系统相机，拍照完将拍完的照片存在sd卡的某一个地方，然后我们调用自己的剪裁工具，不调用系统的剪裁工具了，这样我们剪裁完的图片也就是一张高清的图片了，这样就完美的解决了所遇到的问题。

顺便给大家介绍一款，比较不错的剪裁工具：ucrop

链接：<https://github.com/Yalantis/uCrop>

这款工具，特别的小巧，自定义功能非常强，我在项目中是这样用的：

```
public static String startUCrop(Activity activity, String sourceFilePath,
                                int requestCode, float aspectRatioX, float aspectRatioY) {
    Uri sourceUri = Uri.fromFile(new File(sourceFilePath));
    File outDir = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES);
    if (!outDir.exists()) {
        outDir.mkdirs();
    }
    File outFile = new File(outDir, System.currentTimeMillis() + ".jpg");
    // 裁剪后图片的绝对路径
    String cameraScalePath = outFile.getAbsolutePath();
    Uri destinationUri = Uri.fromFile(outFile);
    // 初始化，第一个参数：需要裁剪的图片；第二个参数：裁剪后图片
    UCrop uCrop = UCrop.of(sourceUri, destinationUri);
    // 初始化UCrop配置
    UCrop.Options options = new UCrop.Options();
    // 设置裁剪图片可操作的手势
    options.setAllowedGestures(UCropActivity.SCALE, UCropActivity.ROTATE, UCropActivity.ALL);
    // 是否隐藏底部容器，默认显示
    options.setHideBottomControls(true);
    // 设置toolbar颜色
    options.setToolbarColor(ActivityCompat.getColor(activity, R.color.colorPrimary));
    // 设置状态栏颜色
    options.setStatusBarColor(ActivityCompat.getColor(activity, R.color.colorPrimary));
    // 是否能调整裁剪框
    options.setFreeStyleCropEnabled(true);
    // UCrop配置
    uCrop.withOptions(options);
    // 设置裁剪图片的宽高比，比如16:9
    uCrop.withAspectRatio(aspectRatioX, aspectRatioY);
    // uCrop.useSourceImageAspectRatio();
    // 跳转裁剪页面
    uCrop.start(activity, requestCode);
    return cameraScalePath;
}
```

11:02



1.35 K/s



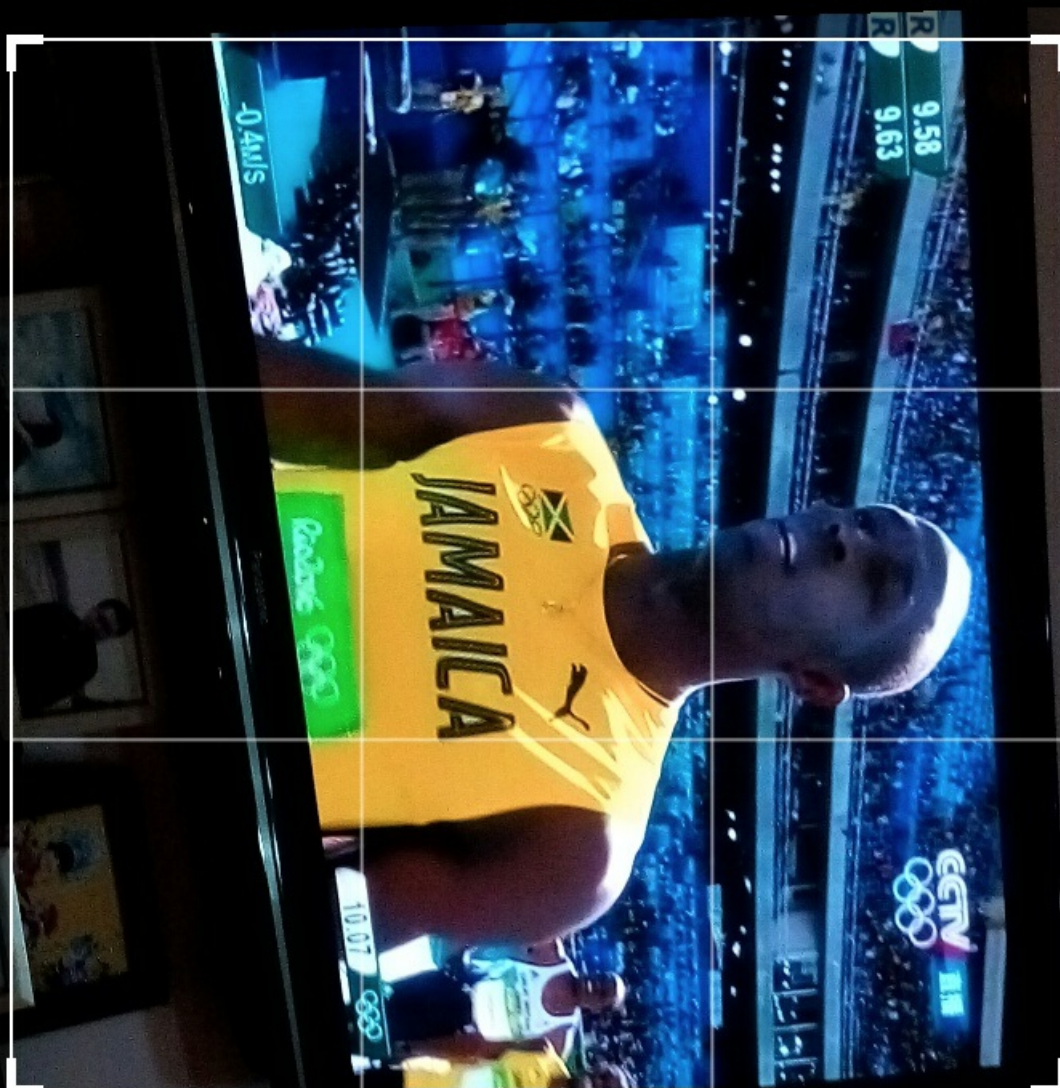
4G
2G



21



裁剪



11:02



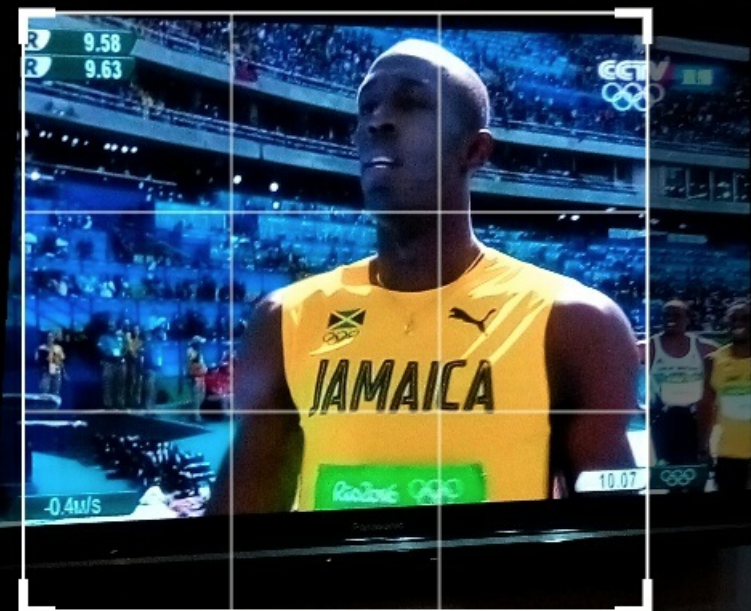
0.09 K/s



21



裁剪



看了图片之后，应该不用我说这个库有多强大了吧，通过简单的设置之后，可以轻松的实现图片的旋转啊，图片的缩放啊，和各种实用的操作，特别的实用。

下面展示一下我们调用系统的拍照，和系统的相册的代码：

```
/**
 * 启动手机相册
 */

private void fromGallery() {
    Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
    intent.addCategory(Intent.CATEGORY_OPENABLE);
    intent.setType("image/*");
    intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(new File(Environment.getExternalStorageDirectory(), "image.jpg")));

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
        startActivityResult(intent, GALLERY_KITKAT_REQUEST);
    } else {
        startActivityResult(intent, GALLERY_REQUEST);
    }
}

/**
 * 启动手机相机
 */

private void fromCamera() {
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    if (FileUtil.hasSdcard()) {
        intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(new File(Environment.getExternalStorageDirectory(), "image.jpg")));
        startActivityResult(intent, CAMERA_REQUEST);
    } else {
        Log.i("sys", "--lin--> SD not exist");
    }
}
```

//如果我们采取的是，调用相册中的照片，我们只需要这样便可以获取到图片，并且直接加载到我们的第三方剪裁库里面。

```
String url = getPath(ReleaseOrderActivity.this, data.getData());
Log.i("lin", "----lin---- url : " + url);
startUCrop(ReleaseOrderActivity.this, url, 1000, 300, 300);
```

//如果我们采用的是调用系统的相机的话，采用这种方式便可以获取到照片，并且加载到我们的剪裁的工具里面啦。

```
startUCrop(ReleaseOrderActivity.this, Environment.getExternalStorageDirectory() + "/" + IMAGE_NAME, 1000, 300, 300);
```

当然，上面我们应用到了一个，`getPath`的方法，当然顾名思义嘛，就是要找到图片的所在位置，但是怕大家懒得自己写，在这里我就给出来：

```
// 以下是关键，原本Uri返回的是file:///...来着的，android4.4返回的是content:///...
@SuppressLint("NewApi")
public static String getPath(final Context context, final Uri uri) {
    final boolean isKitKat = Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT;
    // DocumentProvider
    if (isKitKat && DocumentsContract.isDocumentUri(context, uri)) {
        // ExternalStorageProvider
        if (isExternalStorageDocument(uri)) {
            final String docId = DocumentsContract.getDocumentId(uri);
            final String[] split = docId.split(":");
            final String type = split[0];
            if ("primary".equalsIgnoreCase(type)) {
                return Environment.getExternalStorageDirectory() + "/" + split[1];
            }
        }
        // DownloadsProvider
        else if (isDownloadsDocument(uri)) {
            final String id = DocumentsContract.getDocumentId(uri);
            final Uri contentUri = ContentUris.withAppendedId(Uri.parse("content://downloads/public"), Long.parseLong(id));
            return getDataColumn(context, contentUri, null, null);
        }
        // MediaProvider
        else if (isMediaDocument(uri)) {
            final String docId = DocumentsContract.getDocumentId(uri);
            final String[] split = docId.split(":");
            final String type = split[0];

            Uri contentUri = null;
            if ("image".equalsIgnoreCase(type)) {
                contentUri = Uri.parse("content://media/external/images/media/" + docId);
            }
            return getDataColumn(context, contentUri, null, null);
        }
    }
    // MediaProvider
    else if (isMediaDocument(uri)) {
        final String docId = DocumentsContract.getDocumentId(uri);
        final String[] split = docId.split(":");
        final String type = split[0];

        Uri contentUri = null;
        if ("image".equalsIgnoreCase(type)) {
            contentUri = Uri.parse("content://media/external/images/media/" + docId);
        }
        return getDataColumn(context, contentUri, null, null);
    }
    // Other providers
    else {
        return getDataColumn(context, uri, null, null);
    }
}
```

```

        // Image request type
        contentUri = MediaStore.Images.Media.EXTERNAL_CONTENT_URI;
    } else if ("video".equals(type)) {
        contentUri = MediaStore.Video.Media.EXTERNAL_CONTENT_URI;
    } else if ("audio".equals(type)) {
        contentUri = MediaStore.Audio.Media.EXTERNAL_CONTENT_URI;
    }

    final String selection = "_id=?";
    final String[] selectionArgs = new String[]{
        split[1]
    };
    return getDataColumn(context, contentUri, selection, selectionArgs);
}

// MediaStore (and general)
else if ("content".equalsIgnoreCase(uri.getScheme())) {
    // Return the remote address
    if (isGooglePhotosUri(uri)) {
        return uri.getLastPathSegment();
    }
    return getDataColumn(context, uri, null, null);
}
// File
else if ("file".equalsIgnoreCase(uri.getScheme())) {
    return uri.getPath();
}
return null;
}

public static String getDataColumn(Context context, Uri uri, String selection, String[] selectionArgs) {
    Cursor cursor = null;
    final String column = "_data";
    final String[] projection = {column};

    try {
        cursor = context.getContentResolver().query(uri, projection, selection, selectionArgs, null);
        if (cursor != null && cursor.moveToFirst()) {
            final int index = cursor.getColumnIndexOrThrow(column);
            return cursor.getString(index);
        }
    } finally {
        if (cursor != null) {
            cursor.close();
        }
    }
    return null;
}

/**
 * @param uri The Uri to check.
 * @return Whether the Uri authority is ExternalStorageProvider.
 */
public static boolean isExternalStorageDocument(Uri uri) {
    return "com.android.externalstorage.documents".equals(uri.getAuthority());
}

/**
 * @param uri The Uri to check.
 * @return Whether the Uri authority is DownloadsProvider.
 */
public static boolean isDownloadsDocument(Uri uri) {
    return "com.android.providers.downloads.documents".equals(uri.getAuthority());
}

/**
 * @param uri The Uri to check.
 * @return Whether the Uri authority is MediaProvider.
 */
public static boolean isMediaDocument(Uri uri) {
    return "com.android.providers.media.documents".equals(uri.getAuthority());
}

/**
 * @param uri The Uri to check.
 * @return Whether the Uri authority is Google Photos.
 */
public static boolean isGooglePhotosUri(Uri uri) {
    return "com.google.android.apps.photos.content".equals(uri.getAuthority());
}

```

好了，以上便是我们的解决方案了。

如果大家感觉总结的还不错，可以给我点个赞，或者关注笔者一下~ 有什么问题，也随时欢迎大家留言和我讨论。