自定义view——CameraView

> 作者：https://github.com/linsir6
>
> 文章：http://www.jianshu.com/p/bd91a2b1245d

今天我们我们同样是要做一个自定义view，不过和其他自定义view稍有不同的是，其他的自定view可能继承自的是view，或者viewGroup，今天写的自定义view继承自的是SurfaceView，它具有的功能是，可以直接在view上看到我们自己，并且可以获取到这个流，可以把流编码存储，也可以上传至服务器，当然我这里面只演示了将流中的一帧取出来，然后转乘Bitmap然后加载在了图片上。

既然我们想在view中看到自己，那么不可避免的是我们需要调用系统的相机：

当然这里我比较懒，就把平时常用到的权限全都添加上了，其实如果仅仅想添加相机的权限的话，只需要前几行就可以。

```
<uses-feature android:name="android.hardware.camera"/>
    <uses-feature android:name="android.hardware.camera.autofocus"/>
    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true"/>


    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
    <uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.RECORD_AUDIO"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

相机相关的操作：

```
private Camera mCamera;
private int mCamId = Camera.CameraInfo.CAMERA_FACING_FRONT;

mCamera = Camera.open(mCamId);
Camera.Parameters params = mCamera.getParameters();
Camera.Size size = mCamera.new Size(previewWidth, previewHeight);
//错略的计算相机一帧的大小
mYuvPreviewFrame = new byte[previewWidth * previewHeight * 3 / 2];

params.setPreviewSize(previewWidth, previewHeight);
//设置编码格式
params.setPreviewFormat(ImageFormat.NV21);
// 获取可以对焦的列表
List<String> supportedFocusModes = params.getSupportedFocusModes();
//判断是否有我们想要的，有的话就设置成这个
if (!supportedFocusModes.isEmpty()) {
    if (supportedFocusModes.contains(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE)) {
        params.setFocusMode(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE);
    }
}

mCamera.setParameters(params);

mCamera.setDisplayOrientation(mPreviewRotation);
//回调这个流
mCamera.addCallbackBuffer(mYuvPreviewFrame);
mCamera.setPreviewCallbackWithBuffer(this);
try {
    mCamera.setPreviewDisplay(getHolder());
} catch (IOException e) {
    e.printStackTrace();
}
mCamera.startPreview();
```

surfaceView相关操作：

```
@Override
    public void onPreviewFrame(byte[] data, Camera camera) {
        mPrevCb.onGetYuvFrame(data);
        camera.addCallbackBuffer(mYuvPreviewFrame);
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
    }

    @Override
    public void surfaceCreated(SurfaceHolder arg0) {
        if (mCamera != null) {
            try {
                mCamera.setPreviewDisplay(getHolder());
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder arg0) {
    }
```

到这里为止，我们就可以在view上看到动态的摄像头捕捉的视频了，也可以回调到这个流了，下面就是对流的截取的操作。

将流转换成图片的操作：

```
//设置编码格式
final YuvImage image = new YuvImage(temp, ImageFormat.NV21, 240, 320, null);
ByteArrayOutputStream os = new ByteArrayOutputStream(temp.length);
if(!image.compressToJpeg(new Rect(0, 0, 240, 320), 100, os)){
    return;
}
byte[] tmp = os.toByteArray();
//转换成bitmap
Bitmap bmp = BitmapFactory.decodeByteArray(tmp, 0,tmp.length);

//设置旋转
Matrix matrix = new Matrix();
matrix.setRotate(270f);

Bitmap newBM = Bitmap.createBitmap(bmp, 0, 0, 240, 320, matrix, false);

imageView.setImageBitmap(newBM);
```

项目源码地址：https://github.com/linsir6/mCustomView/tree/master/CameraView

欢迎star～