Glass SDK V0.1.2 使用说明

一、接口说明

com.rokid.glasssdk.GlassControl

public class GlassControl

该类是现实常用的 **Glass** 控制接口和 **glass** 硬件信息读取接口

Constants:

public static final int SHORT_PRESS;

touch 短按事件

public static final int LONG_PRESS;

touchbar 长按事件

public static final int FORWARD_SLIDE;

touchbar 向前滑动

public static final int BACKWARD_SLIDE;

touchbar 向后滑动

Public constructors:

public GlassControl(Context context, UsbDevice dev)

Public methods :

public boolean SetBrightness(**int** value);

设置眼镜亮度值 **rang [0-100]**

public int GetBrightness();

获取当前亮度

public String GetSerialNumber();

获取眼镜序列号

public String GetPCBA();

获取 **PCBA** 编码

public String GetTypeID();

获取设备 **TypeID**

public boolean GetVsyncStatus();

获取显示状态

public String GetOpticalID();

获取光机版本号

com.rokid.glasssdk.OnGlassEvent

public interface OnGlassEvent

该接口用于实现 **glass** 事件回调

Constants：

Public constructors：

Public methods **:**

```
public void OnKeyPress(int keyCode, boolean press)
```
按键事件 **var1:**键值 **var2:**按键状态

```
public void OnTouchPress(int position)
```
返回 **touchbar** 触控当前位置，**bit0-bit7** 分别表示 **8** 个触摸按键 **0** 表示无触摸事件

```
public void OnTochEvent(int event, int value)
```
返回触摸板触发事件 **event** 表示事件编号 **value** 表示滑动长度

```
public void OnImuUpdate(long timeStamp, float Q[])
```
返回 **IMU** 当前 **RotationVectory**

timeStamp：时间戳 （**ms**）

Q：**gameRotationVector**

```
public void OnLsensorUpdate(int lux)
```
返回 **light sensor** 亮度值

```
public void OnPsensorUpdate(boolean status)
```
返回 **PSensor** 状态

二、示例代码

```java
private final BroadcastReceiver usbReceiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if (ACTION_USB_PERMISSION.equals(action)) {
            synchronized (this) {
                UsbDevice device = (UsbDevice) intent.getParcelableExtra(UsbManager.EXTRA_DEVICE);
                if (intent.getBooleanExtra(UsbManager.EXTRA_PERMISSION_GRANTED, false)) {
                    if (device != null) {
                        mTextInfo.setText("Connected!");
                        mGlassCtrl = new GlassControl(context, device);
                        mBrightness.setProgress(mGlassCtrl.GetBrightness());
                        mGlassEvent = new GlassEvent(context, device);
                        mGlassEvent.SetOnGlassEvent(mOnGlassEvent);
                         mHwInfo.setText("SN    :" + mGlassCtrl.GetSerialNumber() + "\n" +
                           "TYPE ID:" + mGlassCtrl.GetTypeID() + "\n" +
                           "PCBA:   " + mGlassCtrl.GetPCBA() + "\n" +
                           "OPTICAL:   " + mGlassCtrl.GetOpticalID() + "\n"

                    }
                } else {
                    Log.d(TAG, "permission denied for device " + device);
                }
            }
        }
    }
};

private OnGlassEvent mOnGlassEvent = new OnGlassEvent(){
    /*四元数转欧拉角*/
   double[] ToEulerAngles(double x, double y, double z, double w) {
      double[] angles = new double[3];
      double sinr_cosp = 2 * (w * x + y * z);
      double cosr_cosp = 1 - 2 * (x * x + y * y);
      angles[0] = Math.atan2(sinr_cosp, cosr_cosp) / Math.PI * 360;
      double sinp = 2 * (w * y - z * x);
      if (Math.abs(sinp) >= 1)
         angles[1] = Math.copySign(Math.PI / 2, sinp) / Math.PI * 360; // use 90 degrees if out of range
      else
         angles[1] = Math.asin(sinp) / 3.14 * 180;
      double siny_cosp = 2 * (w * z + x * y);
      double cosy_cosp = 1 - 2 * (y * y + z * z);
      angles[2] = Math.atan2(siny_cosp, cosy_cosp) /Math.PI * 360;
      return angles;
    }

    @Override
    public void OnKeyPress(int keyCode, boolean press) {
        if(press)
            mTextInfo.setText("Key Press:" + keyCode);
        else
            mTextInfo.setText("Key Releass:"+ keyCode);
    }

    @Override
    public void OnTouchPress(int position) {
        if(position!=0) {
            mTouch.setActivated(true);
            mTouch.setProgress(position);
        }else
            mTouch.setActivated(false);
    }

    @Override
    public void OnTochEvent(int event, int value){
        if(event == TouchEvent.SHORT_PRESS)
            mTextInfo.setText("Touch Event: SHORT");
        else if(event == TouchEvent.LONG_PRESS)
            mTextInfo.setText("Touch Event: LONG");
        else if(event == TouchEvent.BACKWARD_SLIDE)
            mTextInfo.setText("Touch Event: BACKWARD len:" + value);
        else if(event == TouchEvent.FORWARD_SLIDE)
            mTextInfo.setText("Touch Event: FORWARD len:" + value);
    }

    @Override
    public void OnImuUpdate(long timeStamp, float Q[]) {
        double[] angles = ToEulerAngles(Q[0],Q[1],Q[2],Q[3]);
        mTextImu.setText("IMU Data: row:" + (int)angles[0] + "   pitch:" +  (int)angles[1]  +
              "   yaw:" +  (int)angles[2] + "\nts:" + timeStamp);
        //mTextImu.setText("IMU Data:" + Q[0] + "," + Q[1]  + "," +  Q[2] + "," +  Q[3]);
    }

    @Override
    public void OnLsensorUpdate(int lux) {
        mLsensorInfo.setText("LSensor: " + lux);
    }

    @Override
    public void OnPsensorUpdate(boolean status) {
        mPsensorInfo.setText("PSensor:" + status);
    }
};
```