**Qt for Android 使用Honeywell 扫描器的Android SDK开发方法和样例工程**

声明如下：

下面内容属于android网友中的一个转载，但是qt for android是本人亲自调试优化实现并运行成功的demo，属Qt for android原创

1、在工程中导入SDK文件DataCollection.jar。

2、在AndroidManifest.xml中加入下面的代码来获取扫描权限。

<uses-permission android:name="com.honeywell.decode.permission.DECODE" />

3、在Activity类（如MainActivity）的声明中加入BarcodeListener和TriggerListener，以便进行扫描键触发处理和条码事件处理。

[java] view plain copy

public class MainActivity extends AppCompatActivity implements BarcodeReader.BarcodeListener, BarcodeReader.TriggerListener

4、在Activity类（如MainActivity）中定义AidcManager和BarcodeReader对象，通过BarcodeReader对象可以实现扫描属性的设置和扫描功能的使用。

[java] view plain copy

AidcManager manager;

BarcodeReader barcodeReader;

5、在onCreate函数中加入下面的代码，创建扫描对象，设置扫描属性，完成扫描功能的准备工作。

[java] view plain copy

AidcManager.create(this, new AidcManager.CreatedCallback() {

        @Override

        public void onCreated(AidcManager aidcManager) {

            //创建AidcManager和BarcodeReader对象

            manager = aidcManager;

            barcodeReader = manager.createBarcodeReader();

            //设置扫描属性，打开扫描功能

            try {

                barcodeReader.setProperty(BarcodeReader.PROPERTY\_CODE\_128\_ENABLED, true);

                barcodeReader.setProperty(BarcodeReader.PROPERTY\_QR\_CODE\_ENABLED, false);

                barcodeReader.setProperty(BarcodeReader.PROPERTY\_TRIGGER\_CONTROL\_MODE, BarcodeReader.TRIGGER\_CONTROL\_MODE\_CLIENT\_CONTROL);

                barcodeReader.claim();  //打开扫描功能

            } catch(Exception e){

                Toast.makeText(MainActivity.this, "修改属性失败", Toast.LENGTH\_SHORT).show();

            }

            //注册Trigger监听器和Barcode监听器

            barcodeReader.addTriggerListener(MainActivity.this);

            barcodeReader.addBarcodeListener(MainActivity.this);

        }

    });

6、重写onTriggerEvent函数，实现按下扫描键开启补光和扫描，弹起扫描键关闭补光和扫描的功能。

[java] view plain copy

@Override

    public void onTriggerEvent(TriggerStateChangeEvent triggerStateChangeEvent) {

        try {

            barcodeReader.light(triggerStateChangeEvent.getState());//开关补光

            barcodeReader.aim(triggerStateChangeEvent.getState());//开关瞄准线

            barcodeReader.decode(triggerStateChangeEvent.getState());//开关解码功能

        } catch(Exception e){

            Toast.makeText(this, "开关扫描功能失败", Toast.LENGTH\_SHORT).show();

        }

    }

7、重写onBarcodeEvent函数，实现条码数据的接收。

[java] view plain copy

@Override

    public void onBarcodeEvent(BarcodeReadEvent barcodeReadEvent) {

        String barcodeData = barcodeReadEvent.getBarcodeData(); //获取扫描数据

    }

8、重写onFailureEvent函数，实现扫描失败时的处理，也可以不加任何处理，但是必须要有这个函数，否则Build时会报错。

[java] view plain copy

@Override

    public void onFailureEvent(BarcodeFailureEvent barcodeFailureEvent) {

    }

9、在onPause函数中加入barcodeReader.release(); 关闭扫描，在onResume函数中加入barcodeReader.claim(); 恢复扫描。

10、在onDestroy函数中加入下面的代码，关闭所有扫描功能。

[java] view plain copy

barcodeReader.removeTriggerListener(this);

barcodeReader.removeBarcodeListener(this);

barcodeReader.close();

manager.close();

————————————————

版权声明：本文为CSDN博主「Fu\_Lin\_」的原创文章，遵循CC 4.0 by-sa版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/CLinuxF/article/details/79927495