HPRT 蓝牙扫描头 SDK 文档说明

1 扫描头说明

注意：本蓝牙扫描头需要支持 BLE 的蓝牙手机才能使用。

扫描头具有 HID 和 GATT 两种扫描模式。

HID 模式：该模式通过系统连接手机蓝牙后会取代手机的输入法作为手机的输入设备，扫描头采集的数据都可以在文本框里显示。

GATT 模式：该模式需要对接我们的 SDK，通过我们 SDK 开放的接口来获取扫描数据。

2 SDK 使用和接口说明

2.1 SDK 使用

添加 SDK：Android_SCAN_SDKV1.03.01

注册蓝牙服务：

```
<service
android:name="com.example.bluetooth.prt.BluetoothLeService"
android:enabled="true" />
```

添加蓝牙权限：

```
<uses-permission

android:name="android.permission.BLUETOOTH" />

<uses-permission

android:name="android.permission.BLUETOOTH_ADMIN"

/>
```

2.2 接口说明

2.2.1 连接蓝牙

接口名称：

void buleconnect(HidConncetUtil mHidconncetuil,final Context context,onConnect onconnect)

参数：

mHidconncetuil：HidConncetUtil 类的对象。

context:上下文对象。

onconnect：连接反馈接口。

例子：

```
this.mHidConncetUtil = new HidConncetUtil(mContext);
mHprt = HPRTHelper.getHPRTHelper(mContext);
mHprt.setDevice(bluetoothDevice);
mHprt.buleconnect(mHidConncetUtil,mContext,newonConnect(){
@Override
public void succeed() {
// TODO Auto-generated method stub
}
@Override
public void failure() {
// TODO Auto-generated method stub
}});
```

2.2.2 断开蓝牙

接口名称：

void disconnect(onDisconnect ondisconnect)

参数：

ondisconnect：断开连接反馈接口。

例子：

```
mHelper.disconnect(new onDisconnect() {
      @Override
      public void succeed() {
         // TODO Auto-generated method stub
      }
});
```

2.2.3 声音

接口名称：

boolean printSound(int soundstype,int times)

参数：

soundstype：声音类型。

1：长滴。

2：短滴。

times：次数（1-10）。

返回：

true：发送成功。false：发送失败。

例子：

```
if (!hprtHelper.printSound(soundstype, times)) {
   Util.ToastMessage(mContext, "发送失败");
}
```

2.2.4 配置设备

接口名称：

boolean setSetting(final byte[] data)

参数：

data：

0	0x02
1	13
2	长滴时长，取值范围 3-30 ，单位 0.1 秒
3	短滴时长，取值范围 1-5，单位 0.1 秒
4	短滴间隔时长，取值范围 1-5，单位 0.1 秒
5	短滴次数，取值范围 1-10
6	震动时长，取值范围 1-30 ，单位 0.1 秒
7	结尾回车 = 0x00 扫描数据的结尾不添加回车 = 0x01 扫描数据的结尾添加回车
8	大小写字母转换 = 0x00 所有字母保持原样 = 0x01 所有字母转换为小写 = 0x02 所有字母转换为大写
9	短滴和长滴的 duty，取值范围 1-255

10	震动的 duty，取值范围 1-255 （暂未启用）
11	无按键关闭扫描头时间，取值范围 3-60 秒
12	无按键自动关机时间，取值范围 3-60
13	自动连续扫描模式 = 0x00 关闭自动连续扫描模式 = 0x01 开启自动连续扫描模式
14	声音 = 0x00 关闭声音 = 0x01 开启声音
15	校验码，以上所有字节的累加和。

返回：

true：发送成功。false：发送失败。

例子：

参考 Demo 中的 SettingActivity 类。

2.2.5 获取设备参数

接口名称：

void getSettingData(onReadData readdata)

参数：

readdata：反馈接口。

succeed(byte[] data)读取成功。

0	0x81
1	13
2	长滴时长，取值范围 3-30 ，单位 0.1 秒（默认值=5）
3	短滴时长，取值范围 1-5，单位 0.1 秒（默认值=2）
4	短滴间隔时长，取值范围 1-5，单位 0.1 秒（默认值=2）
5	短滴次数，取值范围 1-10（默认值=1）
6	震动时长，取值范围 1-30 ，单位 0.1 秒（默认值=5）
7	结尾回车 = 0x00 扫描数据的结尾不添加回车（默认值=0x00） = 0x01 扫描数据的结尾添加回车
8	大小写字母转换 = 0x00 所有字母保持原样（默认值=0x00） = 0x01 所有字母转换为小写 = 0x02 所有字母转换为大写

9	短滴和长滴的 duty，取值范围 1-255（默认值=50）
10	震动的 duty，取值范围 1-255（默认值=50） （暂未启用）
11	无按键关闭扫描头时间，取值范围 3-60 秒 （默认值=10 秒）
12	无按键自动关机时间，取值范围 3-60（默认值=10 分钟）
13	自动连续扫描模式 = 0x00 关闭自动连续扫描模式（默认值=0x00） = 0x01 开启自动连续扫描模式
14	声音 = 0x00 关闭声音（默认值=0x01） = 0x01 开启声音
15	校验码，以上所有字节的累加和。

返回：

true：发送成功。false：发送失败。

例子：

参考 Demo 中的 SettingActivity 类。

2.2.6 灯光

接口名称：

boolean printLight(int lightstype)

参数：

lightstype：灯光类型。

1：常亮。

2：闪烁。

3：常灭。

返回：

true：发送成功。false：发送失败。

例子：

```
if (!hprtHelper.printLight(stype)) {
    Util.ToastMessage(mContext, "发送失败");
}
```

2.2.7 震动

接口名称：

boolean printShock(int time)

参数：

time：震动时间（1-30 单位 0.1 秒）。

返回：

true：发送成功。false：发送失败。

例子：

```
if(!hprtHelper.printShock(time)){
   Util.ToastMessage(mContext, "发送失败");
}
```

2.2.8 获取版本号

接口名称：

void setReaddeviceversion(onReaddeviceversion onread)

参数：

onread：反馈接口。

data：最后两位代表版本号（低位在前）。

例子：

```
mHelper.setReaddeviceversion(new onReaddeviceversion() {

    @Override
    public void succeed(byte[] data) {
        // TODO Auto-generated method stub
    tv_version.setText("版本号:
          "+data[data.length-1]+data[data.length-2]);
    }
    @Override
    public void failure() {
        // TODO Auto-generated method stub
        tv_version.setText("版本号: 点击获取");
    }
});
```

2.2.9 获取电量

接口名称：

void getElectricity(onElectricity onE)

参数：

onE：反馈接口。

data：

代码	字节数	说明
CMD	1	0x89
DLC	1	3
DATA0	1	电池电压高位，共两字节
DATA1	1	电池电压低位，共两字节
DATA2	1	电池电量百分比
sum	1	效验码

例子：

```
mHelper.getElectricity(new onElectricity() {
    @Override
    public void succeed(byte[] data) {
        // TODO Auto-generated method stub
    tv_electricity.setText("电量: "+data[4]+"%");
    StringparseHex=HPRTHelper.bytetohex(data[2])+
    HPRTHelper.bytetohex(data[3]);
    tv_voltage.setText("电压"
    +Integer.parseInt(parseHex.trim(),16)+"mV");
    }
    @Override
    public void failure() {
    }
    });
```

2.2.10 设置工作模式

接口名称：

boolean setWorkModel(int stype)

参数：

stype：工作模式。

1：GATT

2：HID

返回：

true：发送成功。false：发送失败。

例子：

```
if(mHelper.setWorkModel(GATT)){
   Util.ToastMessage(mContext, "设置成功");
}else{
   Util.ToastMessage(mContext, "设置失败");
}
```

2.2.11 获取工作模式

接口名称：

void getWorkModel(onWorkModel onW)

参数：

onW：反馈接口。

data：

data[2]==1：GATT 模式

data[2]==2：HID 模式

例子：

```
mHelper.getWorkModel(new onWorkModel() {
@Override
    public void succeed(byte[] data) {
        // TODO Auto-generated method stub
        if (data[2]==1) {
            tv_work_model.setText("模式：GATT");
        }else if(data[2]==2){
            tv_work_model.setText("模式：HID");
        }
    }
    @Override
    public void failure() {
        // TODO Auto-generated method stub
    }
});
```

2.2.12 关机

接口名称：

boolean setPowerOff()

返回：

true：发送成功。false：发送失败。

例子：

```
if (!hprtHelper.setPowerOff()) {
   Util.ToastMessage(mContext, "发送失败");
}
```

2.2.13 恢复出厂设置

接口名称：

boolean setRestorefactory()

返回：

true：发送成功。false：发送失败。

例子：

```
if (!hprtHelper.setRestorefactory()) {
   Util.ToastMessage(mContext, "发送失败");
}
```

2.2.14GATT 模式下获取扫描数据

接口名称：

void getGattData(onGattdata onG)

参数：

onG：反馈接口。

data：扫描数据。

例子：

```
helper.getGattData(new onGattdata() {
    @Override
    public void getdata(byte[] data) {
    // TODO Auto-generated method stub
    ed_scandata.setText(Util.byteASCIIstr(data));
    }
});
```

2.2.15 设置震动开关

接口名称：

boolean setShock(int shock)

参数：

shock：是否震动。

1：开启（默认）。

0：关闭。

返回：

true：发送成功。false：发送失败。

例子：

```
if(helper.setShock(shock)){
   Util.ToastMessage(mContext, "修改成功");
}else{
   Util.ToastMessage(mContext, "修改震动失败");
}
```

2.2.16 获取震动开关

接口名称：

void getShock(onShock onS)

参数：

onS：反馈接口。

data：

data[2]==0：关闭

data[2]==1：开启

例子：

```
helper.getShock(new onShock() {
    @Override
    public void succeed(byte[] data) {
    // TODO Auto-generated method stub
    }
    @Override
    public void failure() {
    // TODO Auto-generated method stub
    }
});
```

2.2.17 获取设备名称

接口名称：

void getDeviceName(onDeviceName onD)

参数：

onD：反馈接口。

data：设备名称。

例子：

```
mHelper.getDeviceName(new onDeviceName() {
    @Override
    public void succeed(byte[] data) {
        // TODO Auto-generated method stub
    tv_device_name.setText("设备名称: "
    +Util.byteASCIIstr(data));
    }
    @Override
    public void failure() {
    }
});
```

2.2.18 设置设备名称

接口名称：

void setDeviceName(String deviceName)

参数：

deviceName：设备名称（由字母、数字、空格组成不超过 12 个字）。

例子：

```
if(mHelper.setDeviceName(“HPRT SCAN”){
   Util.ToastMessage(mContext, "修改成功");
}else{
   Util.ToastMessage(mContext, "修改失败");
}
```

2.2.19 暂存功能开启

接口名称：

boolean setTemporarilySave(int TSave)

参数：

TSave： 0：关闭。

1：开启。

例子：

```
if(mHelper.setTemporarilySave(0){
   Util.ToastMessage(mContext, "关闭成功");
}else{
   Util.ToastMessage(mContext, "关闭失败");
}
```

2.2.20 清除暂存数据

接口名称：

boolean cleanTemporarilySaveData()

参数：

无

例子：

```
if(mHelper.cleanTemporarilySaveData(){
   Util.ToastMessage(mContext, "清除成功");
}else{
   Util.ToastMessage(mContext, "清除失败");
}
```

2.2.21 读取暂存统计数据信息

接口名称：

void getTemporarilySaveCount(onTemporarilySaveCount onT)

参数：

isSvaeOpen：暂存功能 true:开启。false：关闭。

savedCount：已存条数。

leftCount：　剩余可存条数。

例子：

```
hprtHelper.getTemporarilySaveCount(new
onTemporarilySaveCount() {
            public void succeed(boolean isSvaeOpen,
int savedCount, int leftCount) {}
            public void failure() {}
});
```

2.2.22 上传暂存数据

接口名称：

void upLodeTemporarilyData(onUpLodeTemporarilyData onUplode)

参数：

data：　　上传的数据。

例子：

```
hprtHelper.upLodeTemporarilyData(new onUpLodeTemporarilyData() {
    public void succeed(byte[] data) {}
    public void failure() {}
});
```