蓝牙型二代身份证阅读及 SIM 开卡接口说明

版本：V1.1

日期: 2015-07-28

适用范围

本方案用于规范蓝牙身份证阅读器设备。

解释权

本方案的解释权属于山东卡尔电气股份有限公司。

版权

本方案的版权属于山东卡尔电气股份有限公司。

修订历史

日期	修订号	描述	著者
2015-07-28	1.0	新建	王旭
2015-08-14	1.1	修改文档标题及名称	王旭

1 SDK 开发包名：idcardreadlib.jar

2 初始化

蓝牙调用类名称定义：BtReaderClient

初始化：BtReaderClient mClient = new BtReaderClient(this);

设置蓝牙连接状态回调：mClient.setCallBack(mCallback);

回调接口名定义：IClientCallBack

在回调接口 onBtState 中显示蓝牙连接的状态：

void onBtState(final boolean is_connect);

Demo：

```
 mClient = new BtReaderClient(this);
 mClient.setCallBack(mCallback);         //设置蓝牙连接状态回调，在回调接口 onBtState 中显示蓝牙连接的状态
public IClientCallBack mCallback = new IClientCallBack(){

                @Override
                public void onBtState(final boolean is_connect){
                     runOnUiThread(new Runnable() {
                          public void run() {

                               Log.d(TAG,"bt_state=" + is_connect);
                               if (is_connect)
                               {
                                    String toast_text = getString(R.string.bt_connected_ts);

                                    Toast.makeText(BTsearchActivity.this, toast_text+devicename,
Toast.LENGTH_SHORT).show();
                                    btstate_view.setText(toast_text+devicename);
                               }
                               else
                               {    Toast.makeText(BTsearchActivity.this,
getString(R.string.bt_lost_connect), Toast.LENGTH_SHORT).show();
                                    btstate_view.setText(R.string.bt_lost_connect);
                               }
                               is_bt_connect = is_connect;
                          }
                     });
                }

                @Override
                public void onIddataHandle(IdCardItem iddata){
```

```
        }
    };
```

3 BtReaderClient 类提供的方法定义：

- 连接蓝牙

  Boolean connectBt(String bt_mac);

  Demo:

```
mClient.connectBt(mac);// 连接设备
```

  bt_mac 为蓝牙 mac 地址　　返回连接结果。

- 设置蓝牙连接状态回调的接口

  void setCallBack(IClientCallBack mCallback);

  mCallback 为回调类 IClientCallBack 的实例对象

  Demo:

```
mClient.setCallBack(mCallback);    //设置蓝牙连接状态回调，在回调接口 onBtState 中显示蓝牙连接的状态
```

- 读身份证信息

  IdCardItem mClient.readIDCard();

  返回身份证信息类 IdCardItem 的一个实例对象，IdCardItem 包含身份证信息的各个属性值，

  IdCardItem idcard = mClient.readIDCard();

  获取姓名：idcard.name

  获取身份证号码：idcard.id_num

  获取性别：getSexStr(idcard.sex_code)

  获取住址：idcard.address

  获取民族：getNationStr(idcard.nation_code)

  出生年：idcard.birth_year

  出生月：idcard.birth_month

  出生日：idcard.birth_day

  发证机关：idcard.sign_office

  起始有效期：年-idcard.useful_s_date_year，月-idcard.useful_s_date_month，日-idcard.useful_s_date_day

  截止有效期：年-idcard.useful_e_date_year，月-idcard.useful_e_date_month，日-idcard.useful_e_date_day

Demo:

```
IdCardItem idcard;
idcard = mClient.readIDCard();
     if (idcard.result_code == 0){       //读取成功
          idcard_view.setText(idcard.name + "\n" +
          idcard.id_num + "\n" +
          idcard.getSexStr(idcard.sex_code) +
          idcard.getNationStr(idcard.nation_code) + "\n" +
          idcard.birth_year +  "-" + idcard.birth_month + "-" + idcard.birth_day + "\n" +
          idcard.address + "\n" +
          idcard.sign_office + "\n" +
     idcard.useful_s_date_year+idcard.useful_s_date_month+idcard.useful_s_date_day + "--" +
          idcard.useful_e_date_year + idcard.useful_e_date_month + idcard.useful_e_date_day);
          img_view.setImageBitmap(idcard.picBitmap);
          img_view.setVisibility(View.VISIBLE);
     }else if (idcard.result_code == 1){  //数据解析失败
          Toast.makeText(mContext, getString(R.string.result_data_err),
Toast.LENGTH_SHORT).show();
                    }
     else if (idcard.result_code == 2){  //读卡超时
     Toast.makeText(mContext, getString(R.string.result_read_overtime),
Toast.LENGTH_SHORT).show();
     }
else if (idcard.result_code == 3){  //蓝牙没有连接
                    Toast.makeText(mContext, getString(R.string.result_bt_no_link),
Toast.LENGTH_SHORT).show();
     }
```

- 断开蓝牙

  Void disconnectBt();

  Demo:

```
mClient.disconnectBt(); // 断开连接
```

- 读 iccid

  String readICCID()

  返回 iccid 卡号或错误信息,错误号详见 demo

  Demo:

```
String result = mClient.readICCID();
if(result==null || result.equals(""))
     return;
byte [] resultByte = hexStringToByte(result);
int resultLen = (int)resultByte[6];
```

```
if(resultLen==1){
      int error = (int)resultByte[7];
      switch (error) {
           case 1:
           Toast.makeText(mContext,"sim 卡未插入", Toast.LENGTH_SHORT).show();
                break;
           case 2:
           Toast.makeText(mContext,"sim 卡插入但是不识别", Toast.LENGTH_SHORT).show();
                break;
           case 4:
           Toast.makeText(mContext,"sim 卡插入但还未成功读取", Toast.LENGTH_SHORT).show();
                break;

           default:
                break;
       }
      }else{
           byte []tmp = new byte[resultLen];
           for(int i=0;i<resultLen;i++){
                tmp[i] = resultByte[7+i];
                }
           idcard_view.setText("长度: "+resultLen +"    卡号 : "+bytesToASCString(tmp));
}
```

- 读 imsi(检查白卡)

  int ReadIMSI(byte szSimType, byte[] szCheckRes, byte[] szImsi2G,
  byte[] szImsiLen2G, byte[] szImsi3G, byte[] szImsiLen3G)

  参数：szSimType // 输入参数 0x00 表示读 2G 卡 0x01 表示读 3G 4G 卡
  szCheckRes // 返回参数 数组长度 1 是否是白卡 0 是，其它 否。
  szImsi2G // 返回参数 数组长度 18 2G 卡 imsi
  szImsiLen2G // 返回参数 数组长度 1 2G 卡 imsi 号的长度
  szImsi3G // 返回参数 数组长度 18 3G 卡的 imsi
  szImsiLen3G // 返回参数 数组长度 1 3G 卡的 imsi 号的长度
  返回的整型数据： 若为 1 读 imsi 成功，否则返回错误码。见 demo

Demo：

```
byte[] szCheckRes = new byte[1];
byte[] szImsi2G = new byte[18];
byte[] szImsiLen2G = new byte[1];
byte[] szImsi3G = new byte[18];
byte[] szImsiLen3G = new byte[1];
// 第一个参数 0x00 表示读 2G 卡   0x01 表示读 3G 4G 卡
int readImsiRes = mClient.ReadIMSI((byte) 0x00, szCheckRes, szImsi2G, szImsiLen2G, szImsi3G,
```

```
szImsiLen3G);
if(readImsiRes!=1){
     int error = readImsiRes;
     switch (error) {
          case 0:
          Toast.makeText(mContext,"imsi 读失败", Toast.LENGTH_SHORT).show();
               break;
          case 2:
          Toast.makeText(mContext,"卡未插入", Toast.LENGTH_SHORT).show();
               break;
          case 3:
          Toast.makeText(mContext,"不识别的 SIM 卡", Toast.LENGTH_SHORT).show();
               break;
          case 4:
          Toast.makeText(mContext,"sim 卡插入但还未初始化", Toast.LENGTH_SHORT).show();
               break;
          default:
               break;
                    }
               }else{
                    StringBuilder sb = new StringBuilder();
                    sb.append("是否白卡 : ");
                    sb.append(szCheckRes[0]==0?"是":"否");
                    sb.append("\n");
                    sb.append("3g: "+bytesToASCString(szImsi3G));
                    sb.append("2g: "+bytesToASCString(szImsi2G));
                    idcard_view.setText(sb.toString());
               }
```

- 写 imsi

  int WriteIMSI(byte[] szImsi1, byte[] szimsi2)

  szImsi1 2Gimsi 卡号　szimsi2　3Gimsi 卡号

  szImsi1　必须不为空，且数据长度为 15;(imsi 前三位 809 已在库中处理)

  szImsi2　可以为 null,此时此卡为 2G 卡。但参数 szImsi2 若不为 Null 时，数据长度必须为 15,(imsi 前三位 809 已在库中处理)。此时此卡为 3G 或 4G。

  返回 iccid 卡号或错误信息,错误号详见 demo

  Demo:

```
byte[] szImsi1 = new byte[15];
for(int i=0;i<15;i++){
     szImsi1[i] = (byte) ((i%10)+0x30); // 例子数据 012345678901234
```

```
        }
        byte[] szImsi2 = new byte[15];
        for(int i=0;i<15;i++){
            szImsi2[i] = (byte) ( (i%10)+0x30);// 例子数据
        }
        if(szImsi1==null || szImsi1.length!=15){// 判断输入参数 1 的格式是否正确。 必须不为空，且数据长度为 15
            Toast.makeText(mContext,"2G 请输入正确的 2G imsi", Toast.LENGTH_SHORT).show();
            return ;
        }
        if(szImsi2!=null&&szImsi2.length!=15){// 判断输入参数 2 的格式是否正确。 可以为 null,此时此卡为 2G 卡。
                                              // 但参数 2 若不为 Null 时，数据长度必须为 15，此时此卡为 3G 或 4G 卡。
            Toast.makeText(mContext,"3G 请输入正确的 3G imsi", Toast.LENGTH_SHORT).show();
            return ;
        }

        int writeImsiRes = mClient.WriteIMSI(szImsi1, szImsi2);
        if(writeImsiRes!=1){
            int error = writeImsiRes;
            switch (error) {
        case 0:
            Toast.makeText(mContext,"imsi 读失败", Toast.LENGTH_SHORT).show();
            break;
        case 2:
            Toast.makeText(mContext,"卡未插入", Toast.LENGTH_SHORT).show();
            break;
        case 3:
            Toast.makeText(mContext,"不识别的 SIM 卡", Toast.LENGTH_SHORT).show();
            break;
        case 4:
            Toast.makeText(mContext,"sim 卡插入但还未初始化", Toast.LENGTH_SHORT).show();
            break;
        default:
            break;
        }
    }else{
        idcard_view.setText("写入 2g: "+bytesToASCString(szImsi1) +"    3g: "+bytesToASCString(szImsi2));
    }
```

- 写短信中心

int WriteSMSC(String szSmsc,byte index)

szSmsc 短信中心号码

Index 短信中心索引，不能为 0，从 1 开始。

返回写入结果。1 代表写入成功否则失败，返回错误码。错误号详见 demo

Demo：

```
byte index = 1;//短信中心号索引（每个 sim 卡可以有几个短信中心线性文件，从 1 开始）
int writeSmscRes = mClient.WriteSMSC(abc,index);
if(writeSmscRes!=1){
     int error = writeSmscRes;
     switch (error) {
          case 0:
          Toast.makeText(mContext,"imsi 读失败", Toast.LENGTH_SHORT).show();
               break;
          case 2:
          Toast.makeText(mContext,"卡未插入", Toast.LENGTH_SHORT).show();
               break;
          case 3:
          Toast.makeText(mContext,"不识别的 SIM 卡", Toast.LENGTH_SHORT).show();
               break;
          case 4:
               Toast.makeText(mContext,"sim 卡插入但还未初始化", Toast.LENGTH_SHORT).show();
               break;
          default:
               break;
          }
     }else{
          idcard_view.setText("写入 smsc： "+abc);
     }
```