NFC 读卡接口

成都鱼住未来科技有限公司

二次开发：http://www.yzfuture.cn
技术支持：faq@yzfuture.cn
售后：sales@yzfuture.cn
电话：028-82880293

日期	版本	说明	作者
2021/07/01	V2.0.0	NFC&OTG 证件读取，支持身份证/港澳居民居住证	TygerZH
2021/12/16	V2.0.1	增加外国人永久居住证	TygerZH

1. 概述

本 SDK 支持二代身份证、护照及 AID 的读取。

二代身份证接口添加了身份鉴权操作，只有当鉴权通过的用户才可以使用公司提供的解码服务器进行身份证解码。

护照及 AID 功能开通需要单独申请。

可兼容目前市场大多数读卡器。

用本 SDK 读证件需要配套我公司专用证件读卡器。

2. 硬件购买

读卡器购买：

https://item.taobao.com/item.htm?id=610814936998

https://item.taobao.com/item.htm?id=611082485024

服务器购买：

https://item.taobao.com/item.htm?id=611230762637

3. 接口概要

接口文件 ReadCardInfoInterface.h 中，同时需要包含 Type.h 文件。

4. 回调

- typedef void(__stdcall *cardReadProgress)(unsigned int nProgress, YZWLHandle nhandle);

  返回身份证读卡进度，一共 20 步。

5. 结构体

- 卡片类型

```
typedef enum cardType
{
 unkwonType = -1,
 ACardType = 0, // A卡
 BCardType = 1  // B卡
};
```

- 证件类型

```
typedef enum _eCardFormatType
{
```

```
 eTwoIDType = ' ', // 身份证
 eTwoGATType = 'J', // 港澳台居民居住证
 eTwoForeignerType = 'I', // 外国人永久居留身份证
}eCardFormatType;
```

- **身份证解码结果**

```
typedef struct TwoIdInfoStructEx
{
 char arrTwoIdName[30];                    //姓名 UNICODE
 char arrTwoIdSex[2];                      //性别 UNICODE
 char arrTwoIdNation[4];                   //民族 UNICODE
 char arrTwoIdBirthday[16];                //出生日期 UNICODE YYYYMMDD
 char arrTwoIdAddress[70];                 //住址 UNICODE
 char arrTwoIdNo[36];                      //身份证号码 UNICODE
 char arrTwoIdSignedDepartment[30];        //签发机关 UNICODE
 char arrTwoIdValidityPeriodBegin[16];//有效期起始日期 UNICODE YYYYMMDD
 char arrTwoIdValidityPeriodEnd[16];       //有效期截止日期 UNICODE YYYYMMDD 有
效期为长期时存储“长期”

 char arrTwoOtherNO[18]; // 通行证类号码
 char arrTwoSignNum[4];      // 签发次数
 char arrTwoRemark1[6]; // 预留区
 char arrTwoType[2];         // 证件类型标识
 char arrTwoRemark2[6]; // 预留区

 char arrTwoIdNewAddress[70];              //最新住址 UNICODE
 char arrReserve[2];                           //保留字节 字节对齐用
 unsigned char arrTwoIdPhoto[1024];        //照片信息
 unsigned char arrTwoIdFingerprint[1024];//指纹信息
 unsigned char arrTwoIdPhotoJpeg[4096];    //照片信息 JPEG 格式
 unsigned int  unTwoIdPhotoJpegLength;//照片信息长度 JPEG 格式
};
```

- **外国人永久居住证**

```
typedef struct _TwoForeignerInfoStruct // 外国人永久居住证
{
 char arrEnName[120];    //英文名
 char arrSex[2];         //性别 UNICODE
 char arrNo[30];         //15 个字符的居留证号码 UNICODE
 char arrNation[6];      //国籍 UNICODE GB/T2659-2000
 char arrCnName[30];     //中文姓名 UNICODE 如果没有中文姓名，则全为 0x0020
 char arrBeginDate[16]; //签发日期 UNICODE YYYYMMDD
 char arrEndDate[16];    //终止日期 UNICODE YYYYMMDD
 char arrBirthday[16];   //出生日期 UNICODE YYYYMMDD
 char arrVersion[4]; // 版本号
```

```
 char arrSignedDepartment[8];//签发机关代码 UNICODE 证件芯片内不存储签发机关
 char arrType[2];       // 证件类型标识
 char arrTwoRemark2[6]; // 预留区
 unsigned char arrPhoto[1024];        //照片信息
 unsigned char arrFingerprint[1024];//指纹信息
}TwoForeignerInfoStruct;
```

- **身份证+港澳台居住证+外国人永久居住证**

```
typedef struct _TwoCardInfoStruct
{
 eCardFormatType etype; // 证件类型，用来区分下面哪个结构有效
 TwoIdInfoStructEx twoInfo; // 身份证或港澳台居住证
 TwoForeignerInfoStruct foreignerInfo; // 外国人永久居住证
}TwoCardInfoStruct;
```

- **护照解码结果**

```
typedef struct _EPassportInfoStruct
{// 护照信息
 char szPaperType[100]; // 证件类型(缩写)
 char szTypeFullName[100];   // 证件类型（全称）
 char szSignedDepartment[100];    // 签发国家或签发机构
 char szENName[100];    // 英文名
 char szCNName[100];    // 中文名
 char szIdNo[100]; // 证件号码
 char szDocumentID[100];// 护照 ID
 char szCountry[100];   // 国籍
 char szBirthday[6];    // 出生日期 UNICODE YYMMDD
 char szSex[1];     // 性别 UNICODE
 char szValidityPeriodEnd[6];     // 有效期截止日期 UNICODE YYMMDD
 char faceImage[1024 * 40]; // 照片信息
}EPassportInfoStruct;
```

6. appkey 申请及终端添加

接口使用过程中需要用到 appkey/secret/userData(终端标识)三个参数，这三个参数的获取，请参照文档《NFC 服务注册流程 V2》创建应用/新增终端步骤进行申请即可。

7. 解码接口

为支持不同语言开发，接口统一为标准 C 接口。

- **cardReadInit**

  SDK 初始化操作，只需要在程序最开始构造的时候调用一次，用来初始化相关初始参数。

- setDeviceType

  设置鱼住标准读卡器还是离线读卡器，默认为鱼住标准读卡器

```
typedef enum _eDeviceType
{
    yzwlType = 0, // 鱼住读卡器
    sdtapiType = 1 // 离线读卡器
}eDeviceType;
```

- loginCardServer

  登录解码服务器。

  szip:中心解码服务器地址（eg:id.yzfuture.cn）

  nport:中心解码服务器开放商品(eg:443)

  szappkey:NFC 应用的标识符，由系统自动生成，且唯一标识某一个 NFC 应用。

  szappSecret:与 appkey 唯一对应的一个串号，与 appkey 配合使用，当用户觉察 appkey 信息泄露的时候，可在后台更新 secret,并在程序中使用新的 secret，此时旧的 secret 失效，即使其它人用旧 secret 也无法使用。

  szAppUserId:终端标识。用户自定义的终端唯一标识符，同一个 NFC 应用下不能重复。只有与终端标识绑定后的设备才有解码授权，否则系统会认为是非法设备，拒绝服务。**（离线读卡器，此处填写离线读卡器 SamId）**

  nerr：错误码

  返回值：

  成功 - true

  失败 - false

- cardOpenDevice

  打开读卡器硬件设备。

  nouttime:超时时间（秒）

  nerr：错误码

  nDeviceNo:读卡器序号，默认为 0（为解决同时有多个相同读卡器的情况，可以选其中某一个为读卡设备）**（如果 setDeviceType 设置为离线读卡器的话，nDeviceNo 应该为 1001~1016 中的某一值）**

  返回值：

  成功 - 读卡器句柄

  失败 <= 0

- setCardType

  设置卡片类型

  nDeviceHandle:打开的设备句柄

  ctype:卡片类型，身份证为 B 卡，一般 IC 卡为 A 卡(A 卡目前只能读 SN，不能进行读写数据操作)

  返回值：

  True - 成功

  False - 失败

```
typedef enum cardType
{
```

```
    ACardType = 0,
    BCardType = 1
  };
```

- cardFindCard

  寻卡

  nDeviceHandle:打开的设备句柄

  bmove:固定写 true

  返回值：

  True - 成功

  False - 失败

- cardGetCardSN

  获取卡片 SN 码**(离线读卡器有可能返回数据为空，视读卡器品牌而定)**

  nDeviceHandle:打开的设备句柄

  szsn:SN 码

  nlen:返回的 SN 码字符串长度，字节为单位(传入时需要赋值 szsn 的字节空间大小，如果小于实际 SN 码长度，则返回失败，空间大小需大于 32 字节)

  返回值：

  True - 成功

  False - 失败

- cardGetCardDN

  获取身份证 DN 码**(离线读卡器返回数据为空)**

  nDeviceHandle:打开的设备句柄

  szsn:DN 码

  nlen:返回的 DN 码字符串长度，字节为单位(传入时需要赋值 szsn 的字节空间大小，如果小于实际 DN 码长度，则返回失败，空间大小需大于 64 字节)

  返回值：

  True - 成功

  False - 失败

- cardGetDeviceNO

  获取读卡器芯片唯一序列号**(离线读卡器返回 SamID,其它品牌读卡器有可能返回数据为空，视读卡器品牌而定)**

  nDeviceHandle:打开的设备句柄

  szsn:芯片唯一序列号

  nlen:返回的序列号字符串长度,字节为单位(传入时需要赋值 szsn 的字节空间大小，如果小于实际序列号长度，则返回失败，空间大小需大于 64 字节)

  返回值：

  True - 成功

  False - 失败

- cardGetDeviceSN

  获取读卡器出厂序列号**(离线读卡器返回 SamID,其它品牌读卡器有可能返**

回数据为空，视读卡器品牌而定）

nDeviceHandle:打开的设备句柄

szsn:读卡器出厂序列号

nlen:返回的序列号字符串长度，字节为单位(传入时需要赋值 szsn 的字节空间大小，如果小于实际读卡器出厂序列号长度，则返回失败，空间大小需大于 64 字节)

返回值：

True - 成功

False - 失败

- cardReadTwoCard

读身份证、港澳居住证或 AID 信息，集成了寻卡、选卡和读身份证的操作。

nDeviceHandle:打开的设备句柄

cardCB:回调的读卡进度

cardinfo:解码后的结果保存在这个结构体里（unicode 格式）

返回值：

True - 成功

False - 失败

- cardReadTwoCardEx

在 cardReadTwoCard 的基础上增加了外国人永久居住证，可读多种证件。

nDeviceHandle:打开的设备句柄

cardCB:回调的读卡进度

cardinfo:解码后的结果保存在这个结构体里（unicode 格式）

返回值：

True - 成功

False - 失败

- cardGetSerialNumber

获取当次读卡的流水号，可用于后期对账**(离线读卡器返回数据为空)**

返回值：

字符串形式的流水号，如果解码在鉴权前就失败，则流水号可能为空

- cardBeep

读卡器蜂鸣一次（10ms）**(离线读卡器有可能无效)**

nDeviceHandle:打开的设备句柄

返回值：

True - 成功

False - 失败

- cardGetLastErrorCode

获取最后一次出错代码

nDeviceHandle:打开的设备句柄

返回值：

True - 成功

False - 失败

- cardGetLastError

  获取最后一次出错信息

  nDeviceHandle:打开的设备句柄

  nlen:返回错误信息的长度

  返回值：

  最后一次出错信息

- cardGetErrorInfo

  根据传入的错误代码查找对应的错误信息

  nDeviceHandle:打开的设备句柄

  nlen:返回错误信息的长度

  nErrorCode:需要查找的错误代码

  返回值：

  查找出错信息

- cardCloseDevice

  关闭读卡器设备

  nDeviceHandle:打开的设备句柄

  返回值：

  无

- decodeCardImage

  将读出来的身份证信息解码成 bmp 格式

  srcimage:读出来的头像信息原数据

  outimage:解码出来的图片数据，内存空间由用户自己管理，不得不于 40K

  outlen:传入时为 outimage 实际大小，传出时为实际图片大小

  返回值：

  True - 成功

  False - 失败

- readEPassportInfo

  读护照信息（此接口功能开通需要单独向本公司申请）

  nDeviceHandle:打开的设备句柄

  szFactoryFlag:用户申请的 appkey(申请过程请和公司商务联系)

  szServerIp:服务器 IP 地址(申请 appkey 时会提供)

  nServerPort:服务器端口(申请 appkey 时会提供)

  szNO:护照号

  szBirth:出生日期

  szEndtime:护照有效期

  cardinfo:解码后的结果保存在这个结构体里

  bTest:固定填 false

  返回值：

  True - 成功

False - 失败

- logoutCardServer
  登出服务器操作

- cardReadUninit
  反初始化操作，最后程序析构的时候调用

8. 错误码

调用 cardGetLastError 获取错误详细信息
或调用 cardGetLastErrorCode 获取错误码

9. 接口调用样例

```
cardReadInit();
TwoIdInfoStructEx cardinfo;
if ("离线读卡器") // 设备型号 YNR201 离线版
{
    setDeviceType(1); //  切换成离线读卡器
    nindex = 1001;
}
else // 设备型号 YNR101 标准版 YNR101N 共享版
{
    setDeviceType(0); // 切换成鱼住标准读卡器
    nindex = 0;
}
int          nerr;
if (loginCardServer("id.yzfuture.cn", 443, "appKey：按自己实际内容填写", "appSecret：按自己实际内容填写",
    "终端标识 userData：按自己实际内容填写", nerr))
{
    YZWLHandle hlHandle = cardOpenDevice(2, nerr, nindex);
    if (hlHandle > 0)
    {
        if (setCardType(hlHandle, BCardType))
        {
            bool     bmove(true);
            if (cardFindCard(hlHandle, bmove))
            {
                bool     bret = cardReadTwoCard(hlHandle, nullptr, cardinfo);
                if (bret)
                {
                    // 解码成功
                    char szBmp[1024 * 40] = { 0 };
```

```
                        int outlen = 1024 * 40;
                        if (decodeCardImage(cardinfo.arrTwoIdPhoto, szBmp, outlen))
                        {
                            // 图片解码成功
                        }
                    }
                }
            }
            cardCloseDevice(hlHandle);
        }
    }
    logoutCardServer();
    cardReadUninit();
```

10. 关于民族代码

民族代码部分对应关系在 sdk->dll->CardInfo.ini 中

11. 关于男女代码

民族代码部分对应关系在 sdk->dll->CardInfo.ini 中

12. 关于外国及地区代码

对应关系在 sdk->dll->CardInfo.ini 中

13.