

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](tel: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、港澳台居民居住证、外国人永久居留身份证。

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。

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. 接口调用样例

```
CardInfoStruct cardinfo;
cardReadInit();
if (鱼住标准读卡器) setDeviceType(0); // 鱼住标准读卡器
else setDeviceType(1); // 离线读卡器
int nerr;
if (loginCardServerEx("id.yzfuture.cn", 443, nerr))
{
    YZWLHandle hlHandle = cardOpenDevice(2, nerr, 0);
    if (hlHandle > 0)
    {
        if (setCardType(hlHandle, BCardType))
        {
            bool bmove(true);
            if (cardFindCard(hlHandle, bmove))
```

```

    {
        bool    bret = cardReadTwoCardEx(hlHandle, nullptr, cardinfo);
        if (bret)
        {
            convertCardInfoToAnsiEx(cardinfo);
            unsigned char*    lpphoto = nullptr;
            if (cardinfo.etype == eOldForeignerType)
            {    // 旧版外国人居住证
                lpphoto = cardinfo.info.foreigner.arrPhoto;
            }
            else if (cardinfo.etype == eNewForeignerType)
            {    // 新版外国人居住证
                lpphoto = cardinfo.info.newForeigner.arrPhoto;
            }
            else if (cardinfo.etype == eTwoGATType)
            {    // 港澳台居住证
            }
            else
            {    // 身份证
                lpphoto = cardinfo.info.twoId.arrPhoto;
            }
            if (lpphoto)
            {
                char szBmp[1024 * 40] = { 0 };
                int outlen = 1024 * 40;
                if (decodeCardImage(lpphoto, szBmp, outlen))
                {
                    // 图片解码成功
                }
            }
        }
    }

    }

    cardCloseDevice(hlHandle);
}

logoutCardServer();
cardReadUninit();

```

5. 回调

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

返回身份证读卡进度，一共 20 步。

6. 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
    eOldForeignerType = 'I', // 外国人永久居留身份证
    eNewForeignerType = 'Y', // 外国人永久居留身份证(新版)
}eCardFormatType;
```

- 身份证和港澳台居住证解码结构

```
typedef struct _TwoIdInfoStruct
{
    char arrName[30];                //姓名 UNICODE
    char arrSex[2];                  //性别 UNICODE
    char arrNation[4];               //民族 UNICODE
    char arrBirthday[16];            //出生日期 UNICODE YYYYMMDD
    char arrAddress[70];             //住址 UNICODE
    char arrNo[36];                  //身份证号码 UNICODE
    char arrSignedDepartment[30];    //签发机关 UNICODE
    char arrValidityPeriodBegin[16]; //有效期起始日期 UNICODE YYYYMMDD
    char arrValidityPeriodEnd[16];   //有效期截止日期 UNICODE YYYYMMDD 有效期
    为长期时存储“长期”

    char arrOtherNO[18]; // 通行证类号码
    char arrSignNum[4];   // 签发次数
    char arrRemark1[6];   // 预留区
    char arrType[2];      // 证件类型标识
    char arrRemark2[6];   // 预留区

    unsigned char arrPhoto[1024];    //照片信息
    unsigned char arrFingerprint[1024]; //指纹信息
};
```

```
}TwoIdInfoStruct;
```

- 外国人永久居住证旧版

```
typedef struct _ForeignerInfoOld // 外国人永久居住证
{
    char arrEnName[120];           //英文名
    char arrSex[2];                //性别 UNICODE
    char arrNo[30];               //15 个字符的居住证号码 UNICODE
    char arrCountry[6];           //国籍 UNICODE GB/T2659-2000
    char arrName[30];             //中文姓名 UNICODE 如果没有中文姓名，则全为
    0x0020
    char arrValidityPeriodBegin[16]; //签发日期 UNICODE YYYYMMDD
    char arrValidityPeriodEnd[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 arrRemark2[6];           // 预留区

    unsigned char arrPhoto[1024]; //照片信息
    unsigned char arrFingerprint[1024]; //指纹信息
}ForeignerInfoOld;
```

- 外国人永久居住证新版

```
typedef struct _ForeignerInfoNew // 外国人永久居住证新版
{
    char arrName[30];             //姓名 UNICODE
    char arrSex[2];               //性别 UNICODE
    char arrNation[4];            //民族 UNICODE
    char arrBirthday[16];         //出生日期 UNICODE YYYYMMDD
    char arrEnName[70];           //外文姓名 UNICODE
    char arrNo[36];               //身份证号码 UNICODE
    char arrSignedDepartment[30]; //签发机关 UNICODE
    char arrValidityPeriodBegin[16]; //有效期起始日期 UNICODE YYYYMMDD
    char arrValidityPeriodEnd[16];  //有效期截止日期 UNICODE YYYYMMDD 有效期
    为长期时存储“长期”

    char arrOtherNO[18]; // 通行证类号码
    char arrSignNum[4];      // 签发次数
    char arrCountry[6];      //国籍 UNICODE GB/T2659-2000
    char arrType[2];         // 证件类型标识
    char arrRemark2[6];      // 预留区
```

```

    unsigned char arrPhoto[1024];          //照片信息
    unsigned char arrFingerprint[1024];    //指纹信息
}ForeignerInfoNew;

```

- 身份证+港澳台居住证+外国人永久居住证

```

typedef struct _CardInfoStruct
{
    eCardFormatType etype; // 证件类型，用来区分下面哪个结构有效
    union
    {
        TwoIdInfoStruct    twoId; // 身份证/港澳台居民居住证
        ForeignerInfoOld foreigner; // 旧版外国人永久居住证
        ForeignerInfoNew newForeigner; // 新版外国人永久居住证
    } info;
}CardInfoStruct;

```

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;

```

- **loginCardServerEx**

登录解码服务器，不使用 appkey 进行管理的话，可直接调用此接口。

离线读卡器可以不调用此接口

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)

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 (解码请尽量使用 cardReadTwoCardEx)**

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，不得大于 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

- **logoutCardServer**
登出服务器操作, **离线读卡器可以不调用此接口**
- **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. 关于民族代码

民族代码部分对应关系在 `sdk->dll->CardInfo.ini` 中

10. 关于男女代码

民族代码部分对应关系在 `sdk->dll->CardInfo.ini` 中

11. 关于外国及地区代码

对应关系在 `sdk->dll->CardInfo.ini` 中