

## 1. 文件结构

**Server:** 身份证解码主程序，负责身份证解码的所有流程

- yzwlReadCardServer.exe: 解码主程序，双击即可运行
- yzwlReadCard\_Server.exe: 解码主程序，以服务形式运行（与 yzwlReadCardServer.exe 同时只可以开一个）
- yzwlServiceAlive.exe: 解码守护服务，与 yzwlReadCard\_Server.exe 互为看守，避免程序退出导致无法解码
- yzwlRCServerSo.dll: 解码服务接口，提供 websocket 和 webapi 两种调用接口，可集成到第三程序或服务中
- 其它 dll: 解码服务需要依赖的动态库
- Config.ini: 解码所需配置文件，如果主程序是服务形式的话需要放在 C 盘根目录或 system32 目录
- websocketclient.html/jquery-1.8.2.js: 测试程序，提供 websocket 和 webapi 两种方式调用
- 创建读卡服务.bat: 双击运行后会在生成一个新的 yzwlReadCard 和 yzwlReadCard 服务进程，开机自动运行，并且循环读卡，无需其它操作。

## 2. 概述

即可以直接使用，也可以进行二次集成后嵌入到其它程序或服务中使用。

可以使用以下两种方式中的任一方式即可：

- 服务方式运行
- 二次集成

## 3. 服务方式运行(两种)

- 直接运行安装包进行安装  
管理员方式运行批处理程序“创建读卡服务.bat”  
等待完成即可，会在系统服务下创建一个 yzwlReadCard 服务，并且开机运行

## 4. 二次集成

加载 yzwlRCServerSo.dll 动态库

- 回调

扩展回调，空实现即可。

```
typedef    bool(__stdcall    *facePhotoDeal)(char*/*[in]*/    szcardInfoJson,
char*/*[out]*/    szerrInfo,    char*/*[out]*/    szuserData,    float&/*[out]*/
fcompareValue);
```

- yzwlRS\_init

初始化操作

facePhotoDeal: 人脸回调

**bcanFace**:是否支持人脸比对（需要在回调中自己处理人脸内容）

**eType**:服务器类型，固定设置为 0

**blocal**:直接写默认值 false 即可

**szDomain**:直接写默认值“com.domain”即可

**返回值**:

True - 成功

False - 失败

- **yzwIrs\_uninit\_init**

反初始化操作

**返回值**:

无

## 5. 二次集成代码演示

```
bool __stdcall onFacePhotoDeal(char**[in]*/ szcardInfoJson, char**[out]*/ szerrInfo,
char**[out]*/ szuserData, float&[out]*/ fcompareValue)
{
    strcpy(szerrInfo, "");
    strcpy(szuserData, "没有数据");
    fcompareValue = 98.0;
    return true;
}

int _tmain(int argc, _TCHAR* argv[])
{
    if (yzwIrs_init(&onFacePhotoDeal, true, eCSServer))
    {
        printf("初始化成功\r\n");
    }
    else
    {
        printf("初始化失败\r\n");
    }

    getchar();
    yzwIrs_uninit();

    return 0;
}
```

## 6. Websocket 接口调用

调用地址: ws://127.0.0.1:30004/ws

- **被动读身份证：**只要客户端连接成功后，不需要进行任何操作。当身份证解码服务解码成功后会给所有连接成功的 **websocket** 客户端分发一次解码信息，客户端解析并显示即可
- **主动读身份证：**按以下格式发送 **json** 字符串，然后接收服务器返回信息解析即可

```
{
  "Cmd": 10000,
  "Head": "YZWL",
  "IPFlag": "MGEyZmU1NmY5ODZlZGM5Ng==",
  "UserParam": "解码信息的 base64 编码",
  "Version": "V1.0.0"
} // 上面为 json 头数据，不需要修改
UserParam 字段的原始格式如下
{
  "AppKey": "99ffb2f98a29071107c7a09ad2c6d096", // 用于设备鉴权的信息
  "DecodePhoto": true, // 是否需要服务端解码身份证图片，如为 false 则需要客户端自己将 wlt 格式转换成 bmp 或其它图片格式
  "FaceCompare": false, // 扩展字段，不用修改
  "PhotoFormat": 1, // 图片解码格式 0-bmp 1-jpg (此处最好设置为 1，因为 bmp 图片过大有可能导致数据出错)
  "ServerIP": "id.yzfuture.cn", // 解码服务器地址，不需要修改
  "ServerPort": 8848 // 解码服务器端口，不需要修改
}
```

- **请求 A 卡 SN（不用修改）**

```
{
  "Cmd": 30400,
  "Head": "YZWL",
  "IPFlag": "YWYyNWMyOWQ1ZTY4ZmJhOQ==",
  "UserParam": "",
  "Version": "V1.0.0"
}
```

- **请求身份证 SN（不用修改）**

```
{
  "Cmd": 20400,
  "Head": "YZWL",
  "IPFlag": "YWYyNWMyOWQ1ZTY4ZmJhOQ==",
  "UserParam": "",
  "Version": "V1.0.0"
}
```

- **服务器返回身份证信息格式**

```
{
  "Cmd": 10001, // 命令编号，不可修改
```

```

"ErrInfo": "", // 错误原因
"Head": "YZWL", // 不可修改
"IPFlag": "OWRkYTkYjliMzQ2MjIzMQ==", // 不可修改
"Ret": 0, // 返回值 0-成功 其它-失败
"UserParam": "解码后的身份证信息 base64 编码", // 身份证信息
"Version": "V1.0.0" // 不可修改
}

```

UserParam 字段的原始格式如下

```

{
  "CardInfo": {
    "Name": "姓名 base64 编码", // unicode + base64
    "Sex": "性别 base64 编码", // unicode + base64
    "Nation": "民族 base64 编码", // unicode + base64
    "Birthday": "出生日期 base64 编码", // unicode + base64
    "Address": "地址 base64 编码", // unicode + base64
    "No": "身份证号 base64 编码", // unicode + base64
    "SignedDepartment": "签发机关 base64 编码", // unicode + base64
    "ValidityPeriodBegin": "开始有效期 base64 编码", // unicode + base64
    "ValidityPeriodEnd": "结束有效期 base64 编码", // unicode + base64
    "OtherNO": "通行证类号码 base64 编码", // unicode + base64
    "SignNum": "签发次数 base64 编码", // unicode + base64
    "Remark1": "备用 1base64 编码", // unicode + base64
    "Type": "证件类型标识 base64 编码", // unicode + base64
    "Remark2": "备用 2base64 编码", // unicode + base64
    "Photo": "身份证照片 base64 编码", // base64
    "Finger": "指纹 base64 编码", //base64
    "SN": "SN 的字符串形式", // 16 进制转字符串
    "DN": "DN 的字符串形式" // 16 进制转字符串
  },
  "FaceInfo": {
    "CompareResult": 0, // 0-成功 -1-设备不支持 其它-失败
    "CompareValue": 0.001, // 相似度
    "ErrInfo": "" // 错误信息
  },
  "OtherInfo": "", // 其它信息, 客户端 so 不负责解析这块, 透传出去, 用户自己负责处理
  "SerialID": "解码流水号",
  "PhotoFormat": 0, // 0-bmp 1-jpg
  "BmpInfo": "图片的 base64 编码"
}

```

- 服务器返回 A 卡 SN

```

{
  "Cmd": 30401, // 命令编号, 不可修改

```

```

"ErrInfo": "", // 错误原因
"Head": "YZWL", // 不可修改
"IPFlag": "OWRkYTkYjliMzQ2MjIzMQ==", // 不可修改
"Ret": 0, // 返回值 0-成功 其它-失败
"UserParam": "base64 编码", // 身份证信息
"Version": "V1.0.0" // 不可修改
}

```

UserParam 字段的原始格式如下

```

{
  "SN": "SN 的字符串形式", // 16 进制转字符串
}

```

- 服务器返回 B 卡 SN

```

{
  "Cmd": 20401, // 命令编号, 不可修改
  "ErrInfo": "", // 错误原因
  "Head": "YZWL", // 不可修改
  "IPFlag": "OWRkYTkYjliMzQ2MjIzMQ==", // 不可修改
  "Ret": 0, // 返回值 0-成功 其它-失败
  "UserParam": "base64 编码", // 身份证信息
  "Version": "V1.0.0" // 不可修改
}

```

UserParam 字段的原始格式如下

```

{
  "SN": "SN 的字符串形式", // 16 进制转字符串
}

```

## 7. Webapi 接口调用

无被动模式, 每次服务器读卡成功后, 需要客户端调用对应的接口才可以获取到解码后的数据。

- 获取身份证信息

接口: <http://127.0.0.1:30004/api/info>

返回: 与 websocket 中 userparam 数据相同

```

{
  "CardInfo": {
    "Name": "姓名 base64 编码", // unicode + base64
    "Sex": "性别 base64 编码", // unicode + base64
    "Nation": "民族 base64 编码", // unicode + base64
    "Birthday": "出生日期 base64 编码", // unicode + base64
    "Address": "地址 base64 编码", // unicode + base64
    "No": "身份证号 base64 编码", // unicode + base64
    "SignedDepartment": "签发机关 base64 编码", // unicode + base64
    "ValidityPeriodBegin": "开始有效期 base64 编码", // unicode + base64
  }
}

```

```

        "ValidityPeriodEnd": "结束有效期 base64 编码", // unicode + base64
        "OtherNO": "通行证类号码 base64 编码", // unicode + base64
        "SignNum": "签发次数 base64 编码", // unicode + base64
        "Remark1": "备用 1base64 编码", // unicode + base64
        "Type": "证件类型标识 base64 编码", // unicode + base64
        "Remark2": "备用 2base64 编码", // unicode + base64
        "Photo": "身份证照片 base64 编码", // base64
        "Finger": "指纹 base64 编码", //base64
        "SN": "SN 的字符串形式", // 16 进制转字符串
        "DN": "DN 的字符串形式" // 16 进制转字符串
    },
    "FaceInfo": {
        "CompareResult": 0, // 0-成功 -1-设备不支持 其它-失败
        "CompareValue": 0.001, // 相似度
        "ErrInfo": "" // 错误信息
    },
    "OtherInfo": "", // 其它信息，客户端 so 不负责解析这块，透传出去，用户自己负责处理
    "SerialID": "解码流水号",
    "PhotoFormat": 0, // 0-bmp 1-jpg
    "BmpInfo": "图片的 base64 编码"
}

```

- 获取 A 卡 SN

接口: <http://127.0.0.1:30004/api/asn>

返回: 与 websocket 中 userparam 数据相同

```

{
    "SN": "SN 的字符串形式", // 16 进制转字符串
}

```

- 获取 B 卡 SN

接口: <http://127.0.0.1:30004/api/bsn>

返回: 与 websocket 中 userparam 数据相同

```

{
    "SN": "SN 的字符串形式", // 16 进制转字符串
}

```

## 8. 问题总结

- 为什么读卡成功后只能读一次身份证信息  
标准读卡器都只能刷一次读一次
- 如何打开测试程序  
双击文件夹中的“websocketclient.html”以浏览器方式打开

websocket 使用 Demo  
(支持被动接收和主动请求两种方式)  
连接到WebSocket 断开WebSocket  
读身份证 读A卡SN 读身份证SN  
连接地址 127.0.0.1:30004  
本demo支持websocket和webapi两种网页调用方式

web api 使用 Demo  
(支持主动请求方式)  
读身份证 读A卡SN 读身份证SN

清空

SN  
姓名 身份证号码  
性别 出生日期 民族  
住址  
签发机关 有效日期 -  
照片

- **读卡服务如何停止**

安装程序安装完成后，在安装目录里面有一个“读卡服务停止.bat”的批处理文件，双击运行即可，然后在服务中查看 yzwlReadCard 和 yzwlServiceAlive 是否是已停止状态。

- **读卡服务如何开启**

安装程序安装完成后，会自动启动服务，以后每次开机都会自动运行。如果手动停止了服务，需要手动开启，开启方式为：在安装目录里面有一个“读卡服务开始.bat”的批处理文件，双击运行即可，然后在服务中查看 yzwlReadCard 和 yzwlServiceAlive 是否是正在运行或已启动状态。

- **用其它鱼住未来公司的读身份证软件，经常打开设备失败**

在服务中查看 yzwlReadCard 和 yzwlServiceAlive 是否是正在运行或已启动状态，如果是请参照上面“读卡服务如何停止”并且服务，然后运行读卡程序。因为服务在占用读卡设备，所以其它软件会经常出现打开设备失败或读卡失败的问题。

- **是否支持离线版读卡器**

支持离线版读卡器。YNR101 和 YZR201 都支持，身份证放于读卡器上即可。