视频连接 WEB 控件使用说明

一、控件简介

名称：　　ICH_RVS_PLAYER.ocx

版本号：　　3.1.1

控件描述：

本 IE 控件实现了视频连接,双向语音,切换码流等功能.

其中 ICH_SysInit, ICH_SysDestroy 是系统基本接口;

属于全局方法，程序生命周期内，只能调用一次;

主要完成了系统初始化，登陆和销毁工作;

其他方法，如 ICH_ConnectStreamer，ICH_DisConnStreamer，ICH_PlayStream 等;

主要完成了采集端连接，播放，控制等功能.

调用流程：

1、 模块初始化 （ICH_SysInit）.

2、 连接采集端 ICH_ConnectStreamer 处理连接响应 ICH_OnConnStreamerStatus.

3、 播放视频 ICH_PlayStream，控制视频（切换码流，静音，对讲等）.

4、 停止视频 ICH_StopStream。

5、 断开采集端（ICH_DisConnStreamer，可选）.

6、 模块销毁 ICH_SysDestroy .

其中 2,3,4,5 步骤是可以循环重复操作的可以多次对视频进行观看停止操作.

控件 IID:　　{A9C0FB7D-07F6-4242-ADD6-71695DFF75E9}

二、 接口说明

SHORT ICH_SysInit (LPCTSTR bstrCompanyId,ULONGLONG ullCompanyKey,LPCTSTR bstrAppId);

/***

*

* Function Name : ICH_SysInit

* Descriptions　: 系统初始化接口

* Input　　: bstrCompanyId: 公司 id;

ullCompanyKey: 公司 key;

bstrAppId:　App id;

以上 3 个参数请到众云视频官网 http://rvs.ichano.cn/download 注册后获取。

* Output　　: N/A

* Return　　: 成功 0，其他失败；

初始化后，可以通过回调接口获取登陆状态：

void ICH_OnSysLoginResult(SHORT sStatus, SHORT sErrid);

sStatus:登陆状态取值如下：

E_RVS_LOGIN_STATE_IDLE　　= 0,

E_RVS_LOGIN_STATE_CONNECTING　= 1,　//连接中...

E_RVS_LOGIN_STATE_CONNECTED　= 2,　//连接成功...

E_RVS_LOGIN_STATE_DISCONNECT　= 3　//连接失败

sErrid:登陆子进度；

typedef enum enPROGRESS{

EN_CBAU_AUTH_PROGRESS_INIT　　= 0,

EN_CBAU_AUTH_PROGRESS_SERVICEGET,

EN_CBAU_AUTH_PROGRESS_SERVICEGETTED,

EN_CBAU_AUTH_PROGRESS_AUTHING,

EN_CBAU_AUTH_PROGRESS_AUTHED,

EN_CBAU_AUTH_PROGRESS_CONNECTING,

EN_CBAU_AUTH_PROGRESS_CONNECTED,

EN_CBAU_AUTH_PROGRESS_REGISTING,

EN_CBAU_AUTH_PROGRESS_REGISTED,

EN_CBAU_AUTH_PROGRESS_ALLOCATING,

EN_CBAU_AUTH_PROGRESS_ALLOCATED,

```
                        EN_CBAU_AUTH_PROGRESS_GETSYSCONFIG,
                        EN_CBAU_AUTH_PROGRESS_UPLOADINFO,
                        EN_CBAU_AUTH_PROGRESS_STARTBUSSINESS,
                        EN_CBAU_AUTH_PROGRESS_STARTED,

                        EN_CBAU_AUTH_PROGRESS_END = 99
                        }EN_AUTH_CONNECT_PROGRESS;

*****************************************************************************/


SHORT ICH_SysDestroy(void);
/*****************************************************************************
* Function Name : ICH_SysDestroy

* Descriptions   : 模块销毁接口；
* Input          : N/A

* Output         : N/A

* Return         : 成功 0，其他失败；
*****************************************************************************/


SHORT ICH_ConnectStreamer(ULONGLONG ullStreamerCid,LPCTSTR bstrUser,LPCTSTR bstrPwd);
/*****************************************************************************
* Function Name : ICH_ConnectStreamer

* Descriptions   : 连接采集端；
* Input          :
                    ullStreamerCid：采集端 cid；
                    bstrUser: 采集端用户名；
                    bstrPwd:    采集端密码；

* Output         : N/A

* Return         : 成功 0，其他失败；

                    连接采集端后，可以通过连接回调接口获取采集端连接状态：
                    void ICH_OnConnStreamerStatus(ULONGLONG ullStreamerCid,
                                                  SHORT sConnStatus)

                    ullStreamerCid:采集端 cid；
                    sConnStatus:采集端在线状态，如下；
```

```
typedef enum enum_RVS_STREAMER_PRESENCE_STATE
{
    E_STREAMER_STATE_INIT = 0,        //初始状态；
    E_TREAMER_STATE_OFFLINE = 1,      //离线；
    E_STREAMER_STATE_ONLINE = 2,      //在线；
    E_STREAMER_STATE_ERRUSERPWD = 3,  //用户名或密码错误；

    E_STREAMER_UNSUPPORTED_DEVICE    = 8, //不支持的设备；
    E_STREAMER_UNSUPPORTED_VERSION   = 9 //不支持的软件版本
}EN_RVS_STREAMER_PRESENCE_STATE;

*****************************************************************************/


SHORT ICH_DisConnStreamer(ULONGLONG ullStreamerCid);
/*****************************************************************************
* Function Name : ICH_DisConnStreamer

* Descriptions  : 断开采集端连接；释放连接资源；
* Input         : ullStreamerCid：采集端 cid；

* Output        : N/A

* Return        : 成功 0，其他失败；
*****************************************************************************/


SHORT ICH_PlayStream(ULONGLONG ullStreamerCid,SHORT sCamId,SHORT sStreamId,SHORT
sMicId,SHORT sPosition)
/*****************************************************************************
*
* Function Name : ICH_PlayStream

* Descriptions  : 发起视频播放请求；
* Input         : ullStreamerCid：采集端 cid；
                  sCamId：摄像头 id，默认取值 0；
                  sStreamId： 流 id，默认取值 0；
                  sMicId：音频麦克风 id，默认 0；
                  sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前
选中的窗口；


* Output        : N/A
```

```
* Return          : 成功 0，其他失败；

                   发起播放请求后，可以通过播放回调接口获取播放是否成功：
                   void ICH_OnPlayStreamStatus(SHORT sPosition, SHORT sStatus)，

                   sPosition：窗口序号；

                   sStatus:播放状态，取值如下；
                   typedef enum enum_RVS_MEDIASTREAM_STATE{
                        EN_RVS_MEDIASTREAM_STATE_CREATED          = 0,//播放成功；
                        EN_RVS_MEDIASTREAM_STATE_FAILED                //播放失败；
                   }EN_RVS_MEDIASTREAM_STATE;
*****************************************************************************/


void   ICH_StopStream(SHORT sPosition);
/*****************************************************************************
*
* Function Name : ICH_StopStream

* Descriptions   : 停止视频流；
* Input           : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output          : N/A

* Return          : 成功 0，其他失败；
*****************************************************************************/


SHORT ICH_GetPlayStatus(SHORT sPosition);
/*****************************************************************************
* Function Name : ICH_GetPlayStatus

* Descriptions   : 获取指定位置的窗口的播放状态，
* Input         : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output          : N/A

* Return          : 当前的播放状态，详见如下定义；
typedef enum enPlayerStatus
{
```

```
    EN_PLAYER_INIT                  = 0x00,
    EN_PLAYER_PREPARING             = 0x01,
    EN_PLAYER_LOADING               = 0x02,
    EN_PLAYER_PLAYING               = 0x03,
    EN_PLAYER_PAUSED                = 0x04,
    EN_PLAYER_STOPING               = 0x05,
    EN_PLAYER_STOPED                = 0x06
}EN_PLAYER_STATUS,*LP_EN_PLAYER_STATUS;
*****************************************************************************/


SHORT ICH_SetMute(SHORT sPosition, SHORT bForceMute);
/*****************************************************************************
*
* Function Name : ICH_SetMute

* Descriptions   : 设置静音；
* Input              : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；
                     bForceMute：1 静音 ，0 取消；

* Output           : N/A

* Return           : 成功 0，其他失败；
*****************************************************************************/


SHORT ICH_IsMute(SHORT sPosition);
/*****************************************************************************
*
* Function Name : ICH_IsMute

* Descriptions   : 是否开启静音；
* Input              : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output           : N/A

* Return           : 1 静音，0 无静音；
*****************************************************************************/


SHORT ICH_SetSpeaking(SHORT sPosition, SHORT bOpenSpeaking);
/*****************************************************************************
```

* Function Name : ICH_SetSpeaking

* Descriptions : 设置对讲；
* Input : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；
bOpenSpeaking：1 打开对讲 ，0 关闭对讲；

* Output : N/A

* Return : 成功开启 0，其他失败；
**********/

SHORT ICH_IsSpeaking(SHORT sPosition);
/**********
* Function Name : ICH_IsSpeaking

* Descriptions : 是否已开启对讲；
* Input : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output : N/A

* Return : 1 开启对讲，0 没有开启；
**********/

SHORT ICH_CheckDbStream(SHORT sPosition);
/**********
* Function Name : ICH_CheckDbStream

* Descriptions : 检测判断是否有双码流；
* Input : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output : N/A

* Return : 1 有双码流，0 没有；
**********/

SHORT ICH_SwitchStream(SHORT sPosition);
/**********
* Function Name : ICH_SwitchStream

* Descriptions : 如果存在双码流，进行码流切换；
* Input : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；

* Output : N/A

* Return : 成功 0，其他失败；
**********/

SHORT ICH_PtzControl(SHORT sPosition, USHORT sCtrlFlag, SHORT sCtrlVal);
/**********
* Function Name : ICH_PtzControl

* Descriptions : 摄像头云台操作方法，目前只支持左右和上下操作，暂不支持缩放；
* Input : sPosition，当前需要操作的窗口序号，0 表示第一个窗口，-1 表示当前选中的窗口；
sCtrlFlag, 参数的取值如下，PTZ_ZOOM 缩放功能暂时不支持；
#define PTZ_UP 1//向上；
#define PTZ_DOWN 2//向下；
#define PTZ_LEFT 3//向左；
#define PTZ_RIGHT 4//向右；
#define PTZ_ZOOM 5//缩放；
sCtrlVal, 取值 1-1000，具体的值可根据厂家参数进行测试调整；
* Output : N/A

* Return : 成功 0，其他失败；
**********/

void ICH_SetWindowsLayout(SHORT sLayoutFlag);
/**********
* Function Name : ICH_SetWindowsLayout

* Descriptions : 设置分屏模式；
* Input : sLayoutFlag 分屏模式，具体的值如下：
//#define LAYOUT_NONE 0
//#define LAYOUT_1X1 1
//#define LAYOUT_2X2 4
//#define LAYOUT_3X3 9
//#define LAYOUT_4X4 16

* Output : N/A

```
* Return            : N/A

*******************************************************************************/


SHORT ICH_GetCurrentWindow(void);
/*******************************************************************************
* Function Name : ICH_GetCurrentWindow

* Descriptions   : 获取当前选中的窗口的位置；
* Input             : N/A

* Output            : N/A
* Return            : 返回当前选中的窗口位置，0 开始；
*******************************************************************************/

void ICH_SetCurrentWindow(SHORT sPosition);
/*******************************************************************************
* Function Name : ICH_SetCurrentWindow

* Descriptions   : 设置选中的指定位置的窗口，0 开始；
* Input             : sPosition,窗口位置，0 开始；

* Output            : N/A
* Return            : N/A
*******************************************************************************/
```