CEC
2.5.0.0.0

座席集成——Openeye H5 软电话接口集成

文档版本 01

发布日期 2024-03-01

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： https://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 修订记录

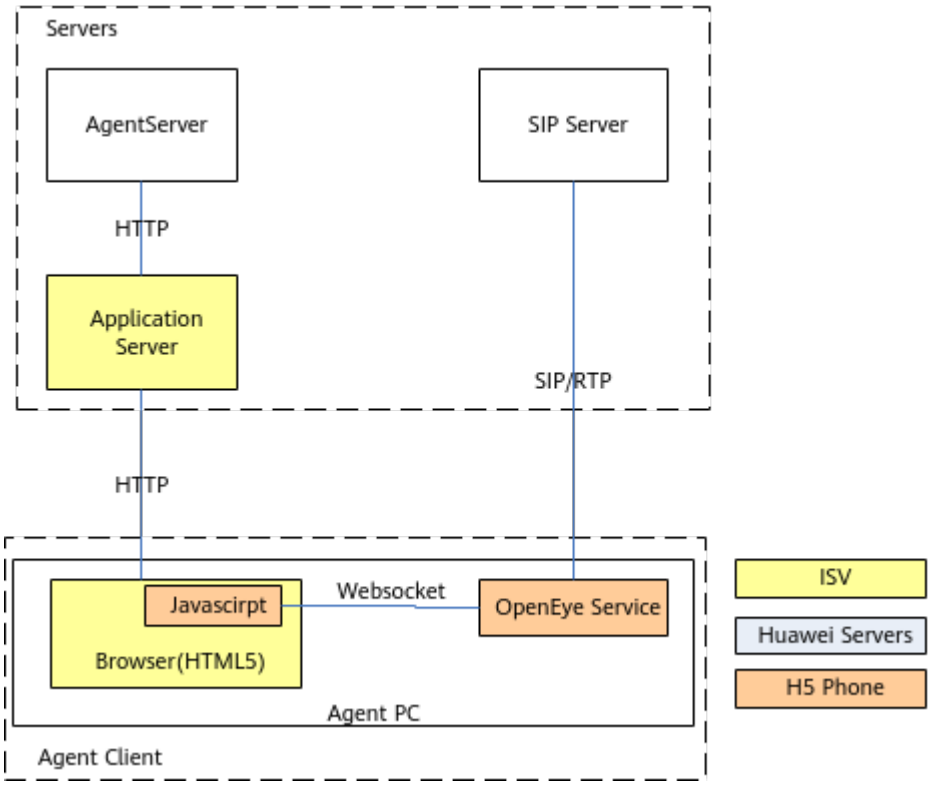
发布日期	文档版本	修订说明
2021-03-08	03	更新了部分接口
2020-07-09	02	添加了屏幕共享相关接口
2019-12-02	01	首次版本文档发布

2 OpenEye H5 软电话接口概述

OpenEye H5软电话支持在网页上实现OpenEye的大部分功能，需要在安装好OpenEye的前提下才能运行，对软硬件的要求与OpenEye相同。

OpenEye H5软电话API接口是基于HTML5（本文简称H5）的javascript接口，可用于控制OpenEye客户端音视频通话组件。

H5软电话接口基于华为OpenEye软件内部组件实现，浏览器集成H5的javascript接口前，需要在本地安装OpenEye客户端软件(参考**3 OpenEye软电话安装指导**)。



3 OpenEye 软电话安装指导

3.1 获取安装包

OpenEye安装包请从AICC的Support路径（联系运维人员获取地址）下获取AICC_***_OpenEye.zip包

在获取到软件包后，需要对软件包的完整性进行校验，操作方法请参考“安装与调测 > 安装指南”中的检查软件包完整性，通过了校验的软件包才能部署。

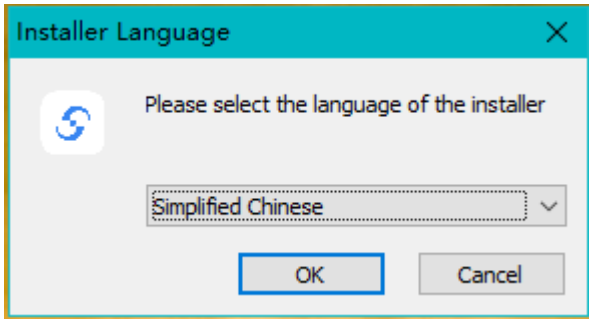
表 3-1 OpenEye 安装包说明

名称	描述
AICC_***_OpenEye.zip	OpenEye Desktop的安装程序压缩包。

3.2 安装过程

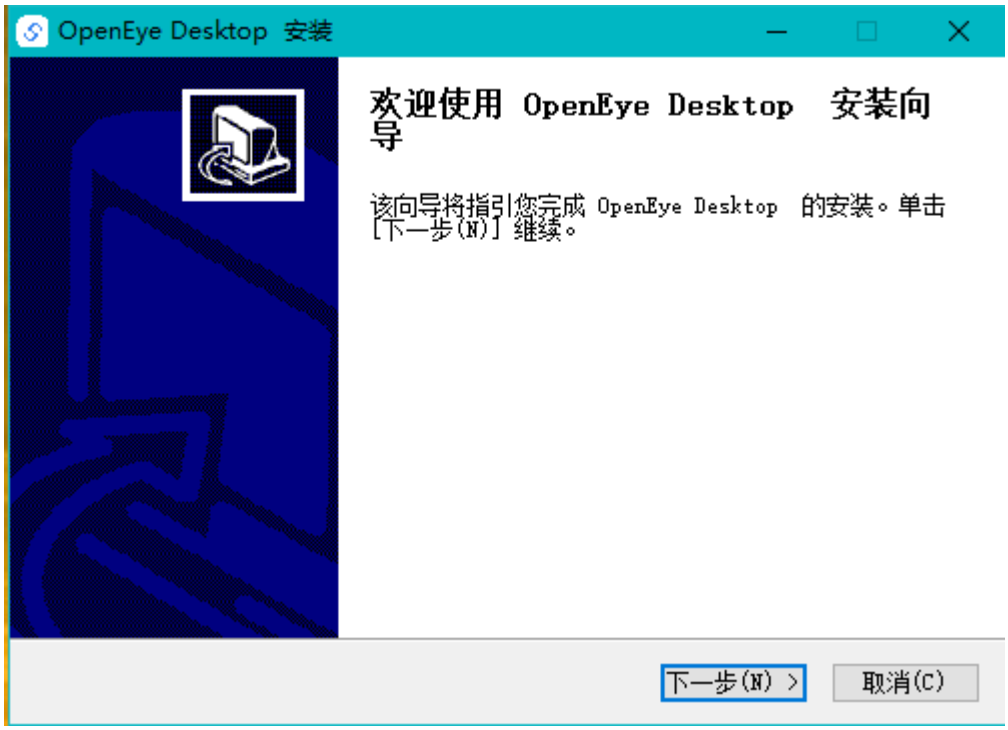
步骤1 解压AICC_***_OpenEye.zip后，双击OpenEyeSetup.exe。

图 3-1 选择安装语言



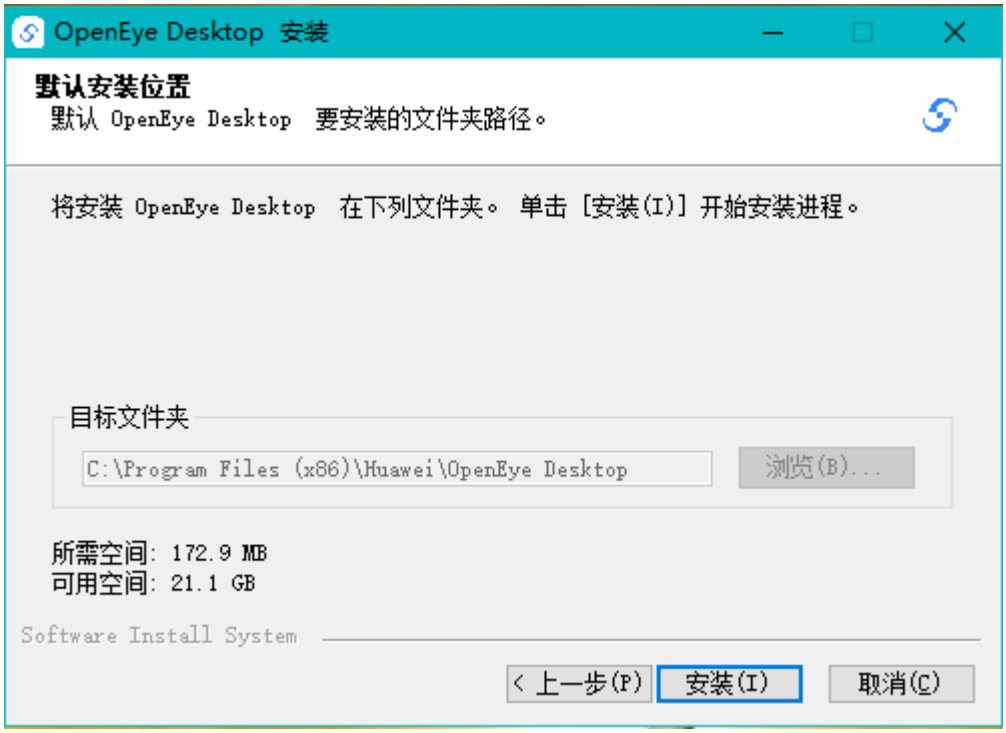
步骤2 单击“OK”。

图 3-2 安装向导



步骤3 点击“下一步”。

图 3-3 安装位置选择

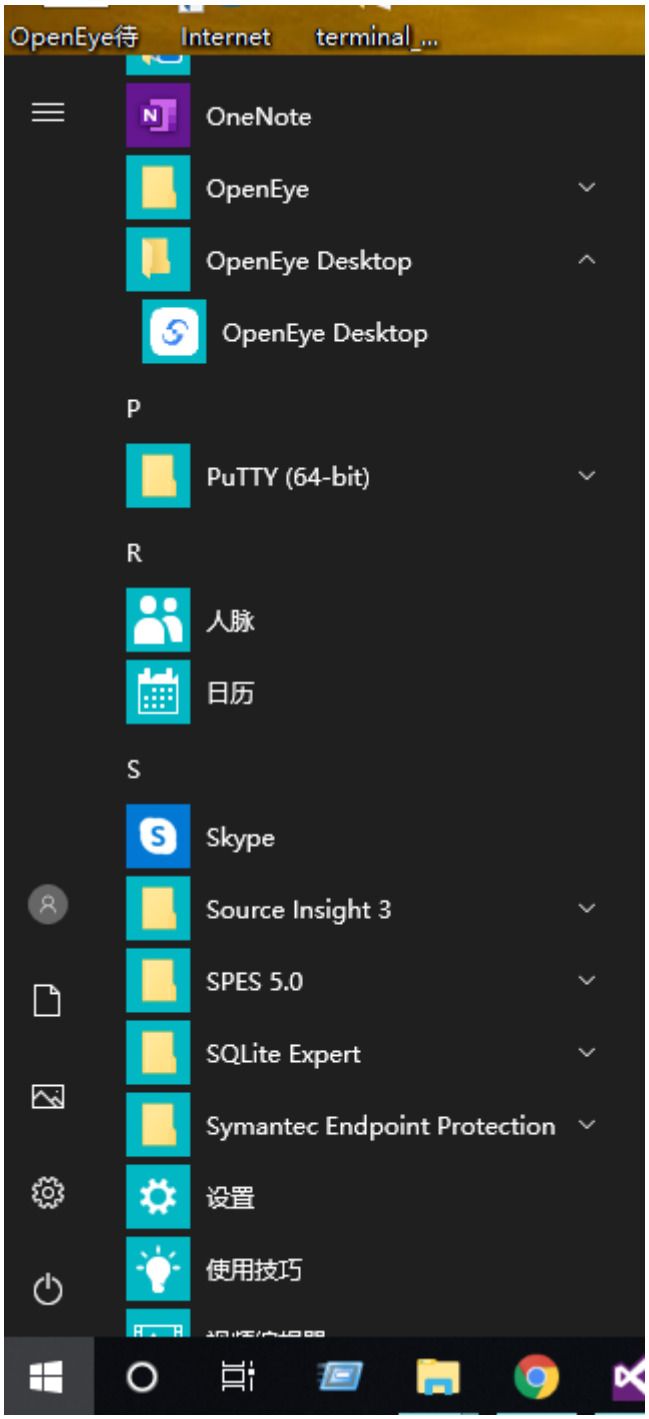


步骤4 点击“安装”，安装完成后显示下图。

图 3-4 安装结果

步骤5 点击“完成”，在操作系统的启动项中，新增安装信息。

图 3-5 软件安装信息



----结束

3.3 验证

前提条件

已经参考**OpenEye软电话安装指导**完成安装并运行。

已经联系研发人员获取了WebDemo工具，下载地址请参见https://bbs.huaweicloud.com/forum/thread-132809-1-1.html。

已经执行了Openeye命令行启动或js启动的命令（即不要弹出Openeye界面的启动方式）：

1. 在程序目录执行命令行启动：
   ```
   ClientApp.exe -mode 2
   ```
2. 网页集成js启动：
   ```
   window.location.href='OpenEyeWebApiShell://-mode,2/';
   ```

测试步骤

步骤1 用文本编辑器打开webdemo压缩包中WebDemo\param.js文件，如下图所示。

图 3-6 param.js

```
param.js
1  var userPhoneNumber; //用户号码
2  var tupCurrentCallType; //当前呼叫类型
3  var tupCurrentCallId = ""; //当前呼叫callId
4  var tupMetuFlag;
5  var tupPlayHandle = -1; //播放句柄
6
7  var OpenEyeConfig = {
8      sipIp: "127.0.0.1",
9      sipPort: 5060,
10     domain: "example.com",
11     localPort: 5060,
12     user_agent: "Huawei OpenEye Desktop For Web"
13 };
14
15 var GlobalSipIp = OpenEyeConfig.sipIp;
16 var GlobalSipPort = OpenEyeConfig.sipPort;
17
18 var EnablePoolMode = false;
19 var OpenEyePoolModeConfig = {
20     sipIp1: "127.0.0.1",
21     sipPort1: 5060,
22     sipIp2: "0.0.0.0",
23     sipPort2: 5060,
24     sipIp3: "0.0.0.0",
25     sipPort3: 5060,
26     sipIp4: "0.0.0.0",
27     sipPort4: 5060
28 };
29
```

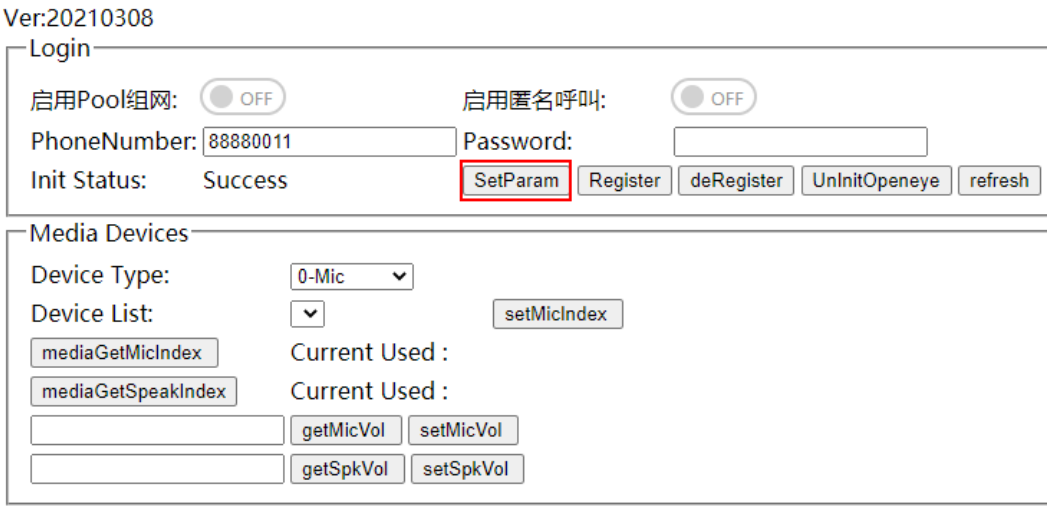
步骤2 修改表3-2中的配置信息并保存，如采用pool模式，则设置EnablePoolMode为true。

表 3-2 修改的配置信息

参数	描述
sipIp	UAP的信令IP地址。
sipPort	SIP服务器的端口号，默认为“5060”。
sipIp1~sipIp4	pool组网UAP的信令IP地址。
sipPort1~sipPort4	pool组网SIP服务器的端口号，默认为“5060”。

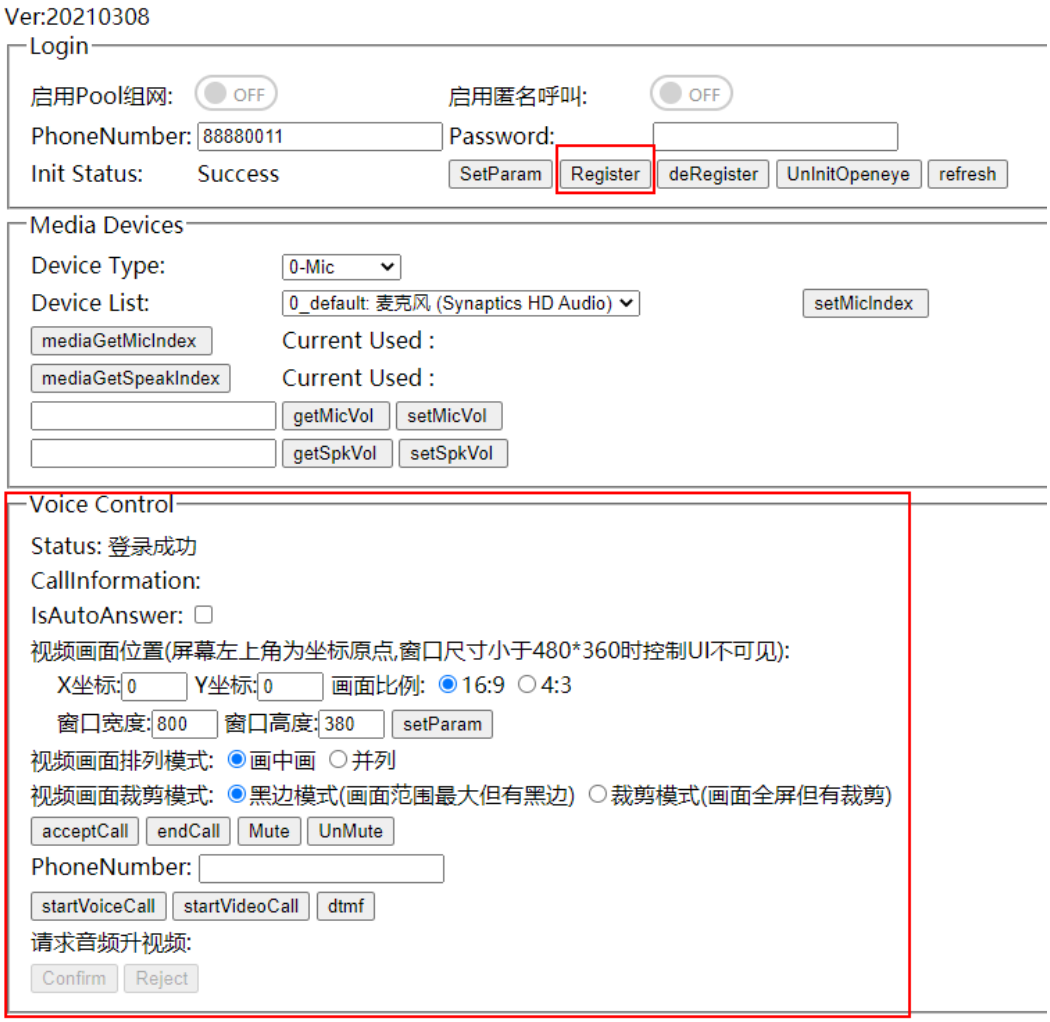
步骤3 用Chrome浏览器打开WebDemo/index.html，显示下图。点击“SetParam”进行基础配置。

图 3-7 index.html



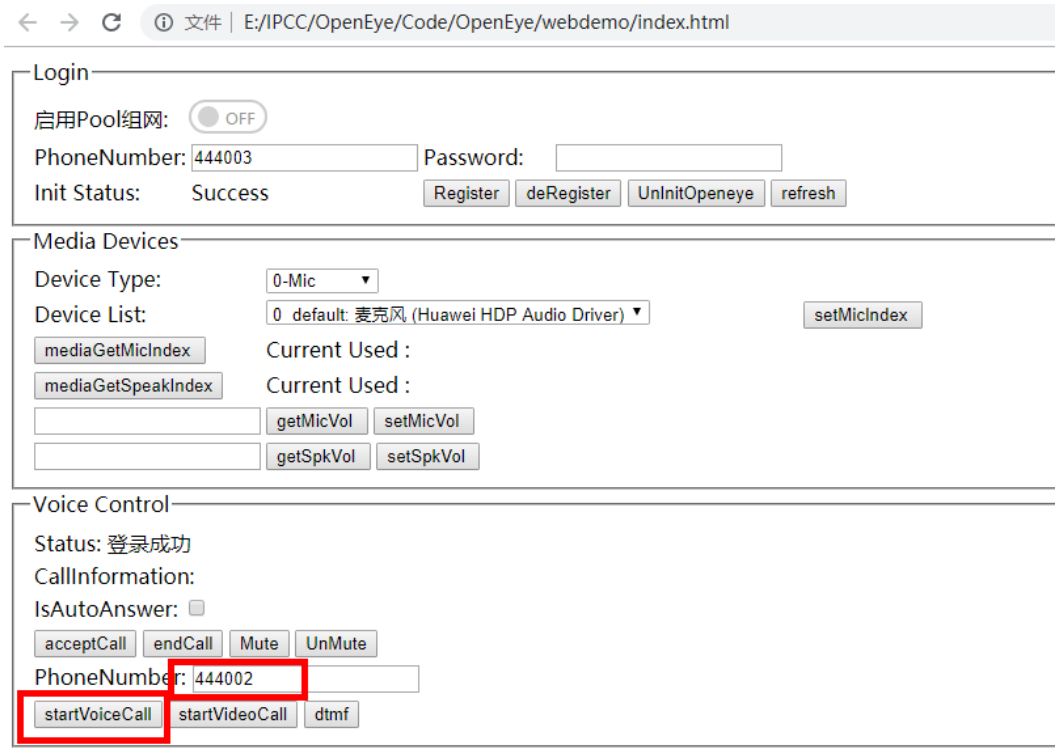
步骤4 在PhoneNumber中输入电话号码，单击“Register”进行注册，注册成功后显示下图。

图 3-8 注册结果



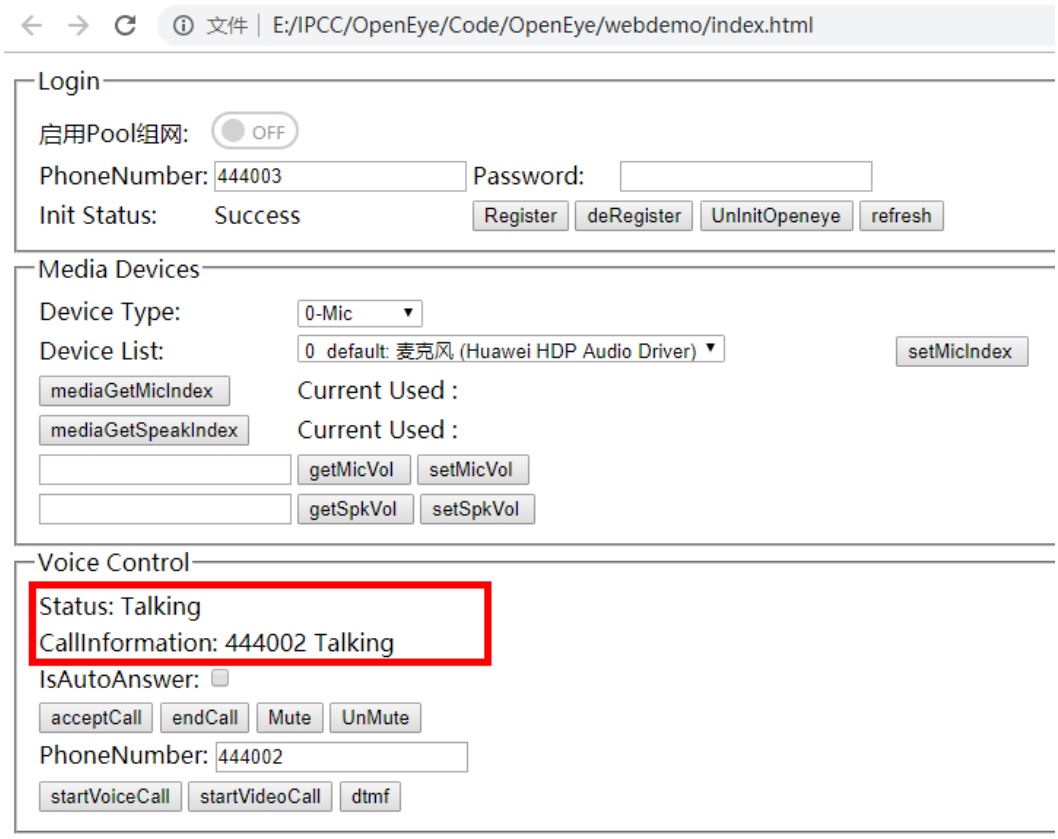
步骤5 在Voice Control下的PhoneNumber输入另外一个已经注册USM的话机号码（即在该组网环境中已注册的软电话号码，如444002），并点击startVoiceCall，如下图所示。

图 3-9 外呼操作



步骤6 444002应答后，呼叫建立，显示如下图所示。

图 3-10 外呼结果

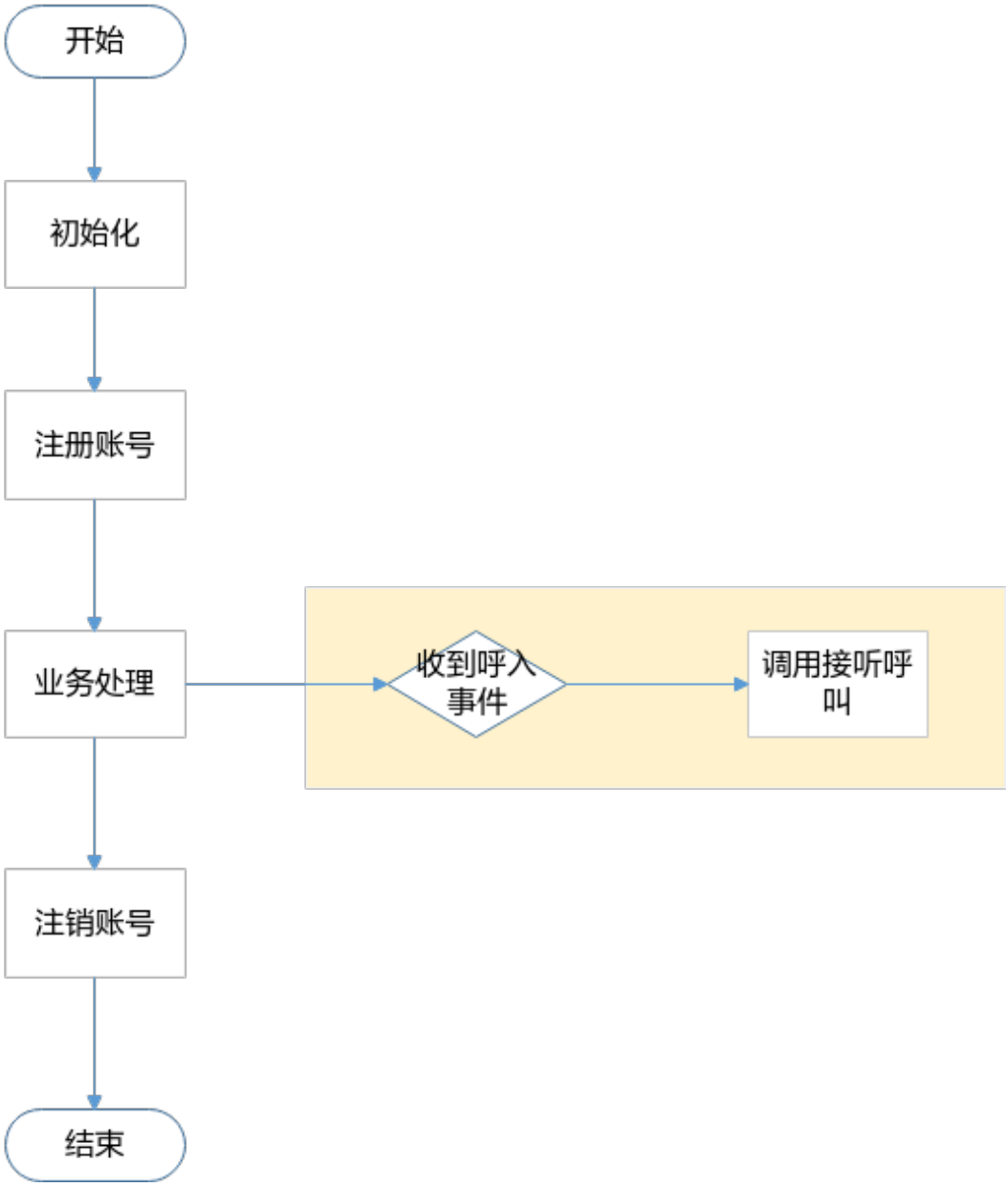


----结束

4 座席侧集成 H5 软电话开发指导

4.1 业务开发总体流程

您在使用OpenEye提供的H5软电话接口集成开发时，基本的流程步骤包含：初始化业务组件、注册帐号、业务处理、注销用户帐号。



流程说明

1. **初始化组件**：实现对业务组件进行资源初始化，设置第三方应用程序的全局业务配置参数。
2. **注册帐号**：调用相应接口完成向服务器的SIP注册。
3. **业务处理**：收到呼入事件后，由第三方应用自动调用接听呼叫接口。
4. **注销帐号**：实现用户退出，确保业务接口使用的安全性（若不注销账号，则OpenEye会保持该账号一直处于已注册状态）。

4.2 Sample Codes

Sample Codes下载路径：暂无

如需获取示例代码，请联系OpenEye开发运营团队，获取Sample Code及技术支持。

名称	描述
WebDemo	基于OpenEye Desktop的网页端Demo，包含音视频呼叫、设备管理等功能。仅限于参考，不建议直接使用，请自行开发适配网页。

4.3 初始化

应用场景

在座席侧使用Openeye软电话接口进行注册话机号码时，需要完成组件初始化。

前提条件

1. 已经完成WebDemo的下载。
2. 已经安装Openeye的本地程序包，并且OpenEye的本地程序已经启动。
3. 已经完成param.js中服务器等信息的配置。

流程说明

说明

Demo中代码路径：WebDemo\js\OpenEye_SDK.js。

步骤1 初始化SDK。

说明

打开webdemo页面，会收到OpenEye返回的消息，只有收到onOpeneyeLoginReady后才能调用OpeneyeLogin下的接口，只有收到onOpeneyeCallReady后才能调用OpeneyeCall下的接口。

```
function initOpeneye(){
    if (global_openEye_SDK== null)
    {
       global_openEye_SDK = new OpenEye_SDK({
        onOpeneyeDeamonReady: onOpeneyeDeamonReady,
        onOpeneyeDeamonClose: onOpeneyeDeamonClose,
        serviceStartUp: serviceStartUp,
        serviceShutDown: serviceShutDown,
        onOpeneyeLoginReady: onOpeneyeLoginReady,
        onOpeneyeLoginClose: onOpeneyeLoginClose,
        onOpeneyeCallReady: onOpeneyeCallReady,
        onOpeneyeCallClose: onOpeneyeCallClose,
        onVersionInfoNotify : onVersionInfoNotify
    });

    }
}
function onVersionInfoNotify(data)
{
    writeLog("version is " + data.param.version);
}
function onOpeneyeDeamonReady() {
    writeLog("OpenEye Deamon is Ready");
}

function onOpeneyeDeamonClose() {
    writeLog("OpenEye Deamon is Closed, please restart OpenEye Deamon it.");
```

```
    global_openEye_SDK= null;
}


function serviceStartUp() {
    writeLog("Openeye Service StartUp");
}

function serviceShutDown() {
    writeLog("Openeye Service is shutdown, please restart it.");
}

function onOpeneyeLoginReady(){
    writeLog("onOpeneyeLoginReady");
}


function onOpeneyeLoginClose(){
    writeLog("onOpeneyeLoginClose");
}


function onOpeneyeCallClose() {
    writeLog("onOpeneyeCallClose");
}


function onOpeneyeCallReady() {
    writeLog("onOpeneyeCallReady");
}
```

步骤2 初始化OpenEyeCall的参数和监听呼叫事件。

```
function initOpeneyeCall(localIp)
{
    console.info("onOpeneyeCallReady");
    initOpeneyeCall();
}
```

- 调用**setBasicCallEvent**接口监听的OpenEyeCall的呼叫事件。

```
function initOpeneyeCall()
{
    global_openEye_SDK.openEyeCall.setBasicCallEvent({
        onCallIncoming: onCallIncoming,
        onCallOutGoing: onCallOutGoing,
        onCallRingBack: onCallRingBack,
        onCallConnected: onCallConnected,
        onCallEnded: onCallEnded,
        onCallEndedFailed: onCallEndedFailed,
        onCallRtpCreated: onCallRtpCreated,
    onCallOpenVideoReq: onCallOpenVideoReq
    });
}
```

- 调用**config**接口设置OpenEyeCall的SIP服务器信息。

  说明

  必须保证SIP服务器的信息填写正确，否则无法进行账号注册

```
function sipBasicCfg() {
    global_cloudIPCC_SDK.openEyeCall.config({
        networkInfo: {
            serverAddr: GlobalSipIp,
            sipServerPort: GlobalSipPort,
            sipTransportMode: 0,
            httpPort: 0
        },
    }, {response: configResponse});
}
```



```
function configResponse(data)
{
    if (data.result == 0) {
        writeLog("Set OpenEyeCall Sip Config Success.");
        register();
    } else {
        writeLog("Set OpenEyeCall Sip Config Failed.");
    }

}
```

----结束

4.4 账号注册

应用场景

注册座席的话机。

前提条件

已经完成OpenEyeCall参数的初始化，并且SIP服务器信息填写正确。

流程说明

调用OpenEyeCall的**register**接口进行接听呼叫。

说明

- 调用注册接口除了需要处理返回结果，还需要监听**onRegStatusUpdate**事件。
- 注册是一个异步的过程，并不是注册返回结果是0就表示注册成功，还需要看onRegStatusUpdate的返回参数register_state是否为3，只有为3才表示成功。

```
/**
 * 注册
 */
function register() {
    var phoneNumber = document.getElementById("phoneNumber").value;
    var password = document.getElementById("password").value;
    userPhoneNumber = phoneNumber + "@" + GlobalSipIp;//OpenEyeConfig.domain;
    console.info("register info="+userPhoneNumber);

    var sipMode=1;
    if(EnablePoolMode){
        sipMode=1;
    }
    this.global_openEye_SDK.openEyeCall.register(userPhoneNumber, phoneNumber, password, sipMode, {
        onRegStatusUpdate: onRegStatusUpdate,
        onForceUnReg: onForceUnRegInfo,
        response: registerResponse
    });
}
/**
 * 注册结果
 * @param {*} data
 */
function registerResponse(data) {
   if (data.result == 0) {
        console.info("Register Operation Success");
    getMediaDevices();
    } else {
        console.error("Register Operation Failed");
```



```
        console.error(data);
    }

}

/**
 * 注册结果上报
 * @param {*} data
 */
function onRegStatusUpdate(data) {
    var userNumber = data.param.user_number;
    var state = ["unregister", "registering", "deregistering", "registered", "deregistered"];
    var reason = data.param.reason_code; //complete reason code refer to: http://blog.csdn.net/
kafeiwuzhuren/article/details/7242791
    if (reason == 403) {
        console.log("403:forbidden");
    }
    if (reason == 408) {
        console.log("408:request overtime");
    }
    var currentState = state[data.param.register_state];
    var obj = { userNum: userNumber, stateInfo: currentState, reasonInfo: reason }
    console.log("onRegStatusUpdate");
    console.log(data);
    document.getElementById("phoneStatus").innerText = obj.stateInfo;
    if (data.param.register_state == 3) {
        document.getElementById("voiceControlDiv").style.visibility = "visible";
    }
    if(data.notify==1004){
        document.getElementById("phoneStatus").innerText = "登录成功";
        document.getElementById("voiceControlDiv").style.visibility = "visible";
    }

}

function onForceUnRegInfo(data) {
    document.getElementById("phoneStatus").innerText = "DeRegister";
    document.getElementById("voiceControlDiv").style.visibility = "hidden";
    userPhoneNumber = "";
    tupCurrentCallId = "";
    console.log("onForceUnRegInfo");
    console.log(data);

}
```

4.5 接听呼叫

应用场景

座席侧收到OpenEye上报的呼入事件后，自动调用接听呼叫接口，实现来话的接听。

前提条件

- 座席已经签入CTI平台。
- 收到**onCallIncoming**。

流程说明

调用OpenEyeCall的**acceptCall**接口进行接听呼叫。

```
/**
 * 来电事件
 **/
function onCallIncoming(data) {
```

```
    //记录一下来电的callid，后续接听接口需要用到该参数
    var tupCurrentCallId = data.param.call_id;

    //此处调用仅是演示如何调用接听呼叫接口，实际开发中请根据需要在合适地方调用该接口
    acceptCall(tupCurrentCallId);
}
/**
 * 接听呼叫
 */
function acceptCall(tupCurrentCallId) {
   if (tupCurrentCallStatus == OPENEYE_CALL_STATUS.ALERTING) {
        this.global_openEye_SDK.openEyeCall.acceptCall(tupCurrentCallId, tupCurrentCallType, { response:
onAcceptCallReponse });
    } else {
        console.error("Phone status is invalid. Now it's " + tupCurrentCallStatus);
        alert("Phone status is invalid. Now it's " + tupCurrentCallStatus);
    }
}
/**
 * 接听接口的响应
 */
function onAcceptCallReponse(data) {
    if (data.result == 0) {
        console.error("AcceptCall success. ");
    } else {
        console.error("AcceptCall failed. The ErrorCode is " + data.result);
        console.info(data);
        alert("AcceptCall failed.  The ErrorCode is " + data.result);
    }
}
```

4.6 拨打呼叫

应用场景

座席侧登陆成功后，主动调用呼叫接口，实现音视频电话的主动呼叫。

前提条件

- 座席已经签入CTI平台。
- 话机账号登陆注册成功。

流程说明

调用OpenEyeCall的**startCall**接口进行接听呼叫。

```
/**
 * 呼出
 */
function startCall() {
    //需要区分匿名呼叫还是非匿名
    var ischecked = document.getElementById("toggle-button-anonymous").checked;
    if(ischecked){

this.global_openEye_SDK.openEyeCall.startAnonymousCall(document.getElementById("calloutNumber").valu
e, false, {
            response: startCallResponse
        });
    }else{
        this.global_openEye_SDK.openEyeCall.startCall(document.getElementById("calloutNumber").value,
false, {
            response: startCallResponse
        });
    }
```



```
}

/**
 * 呼出的响应
 */
function startCallResponse(data) {
    if (data.result == 0) {
        console.info("StartCall success. callid="+JSON.stringify(data));
        tupCurrentCallId = data.param.callId;
    } else {
        console.error("StartCall failed. The ErrorCode is " + data.result);
        console.info(data);
        alert("StartCall failed.  The ErrorCode is " + data.result);
    }
}
```

4.7 结束呼叫

应用场景

座席侧登陆成功后，在来电，去电或者通话过程中，主动调用结束呼叫接口，实现音视频呼叫的挂断。

前提条件

- 座席已经签入CTI平台。
- 话机账号登陆注册成功。
- 处于呼叫或者通话状态中。

流程说明

调用OpenEyeCall的**endCall**接口进行接听呼叫。

4.8 账号注销

应用场景

座席从CTI平台签出后，也需要注销话机账号。

前提条件

- 座席已经签入CTI平台。
- 账号已经注册。

流程说明

调用OpenEyeCall的**deRegister**接口进行注销。

```
/**
 * 注销
 */
function deRegister() {
    if (global_openEye_SDK!= null && global_openEye_SDK.openEyeCall!= null)
    {
        global_openEye_SDK.openEyeCall.deRegister(global_userPhoneNumber, {
```



```
            response: deRegisterResponse
        });
    }
}
/**
 * 注销结果
 * @param {*} data
 */
function deRegisterResponse(data) {
    if (data.result == 0) {
        writeLog("Phone DeRegister Success.");
    } else {
        writeLog("Phone DeRegister Failed.");
    }
}
```


5 音频呼叫接口

5.1 初始化

5.1.1 OpenEye_SDK(创建对象，并初始化)

接口描述

初始化SDK，会在内部实现与OpenEyeDeamon，OpenEyeLogin，OpenEyeCall三个模块的WebSocket连接。

注意事项

- OpenEye的本地客户端已经启动。
- 每台电脑上只能有一个网页进行一次初始化SDK的操作。
- 第三方应用页面已经加载OpenEye_SDK.js。

方法定义

```
function OpenEye_SDK(opts)
```

参数描述

表 5-1 参数说明

参数名	类型	可选/必选	描述
opts	**Opts**	必选	回调方法。

表 5-2 Opts

参数名	类型	可选/必选	描述
onOpeneyeDeamonReady	function	必选	表示与OpenEye客户端的WebSocket连接建立。
onOpeneyeDeamonClose	function	必选	表示与OpenEye客户端的WebSocket连接关闭。 **说明** 如果与OpenEye客户端的WebSocket连接关闭，则与OpenEyeCall和OpenEyeLogin的WebSocket连接也会关闭。
serviceStartUp	function	必选	表示本地OpenEye服务已经启动。 **说明** 只有本地OpenEye服务启动后，才能与OpenEyeCall和OpenEyeLogin的WebSocket连接建立。
serviceShutDown	function	必选	表示本地OpenEye服务已经关闭。
onOpeneyeLoginReady	function	必选	表示与OpenEyeLogin的WebSocket连接建立。
onOpeneyeLoginClose	function	必选	表示与OpenEyeLogin的WebSocket连接关闭。
onOpeneyeCallReady	function	必选	表示与OpenEyeCall的WebSocket连接建立。
onOpeneyeCallClose	function	必选	表示与OpenEyeCall的WebSocket连接关闭。
onVersionInfoNotify	function	必选	版本信息通知。

使用示例

```
function onOpeneyeDeamonReady() {
    console.info("Openeye Deamon is Ready");
}

function onOpeneyeDeamonClose() {
    console.error("Openeye Deamon is Closed,please restart it");
    global_openEye_SDK = null;
}
```



```
function serviceStartUp() {
    console.info("OpenEye Service StartUp");
}

function serviceShutDown() {
    console.error("OpenEye Service is shutdown,please restart it");
}

function onOpeneyeCallClose() {
    console.error("onOpeneyeCallClose");
}


function onOpeneyeCallReady() {
    console.info("onOpeneyeCallReady");
}

function onOpeneyeLoginReady() {
    console.info("onTupLoginReady");
}


function onOpeneyeLoginClose() {
    console.info("onOpeneyeLoginClose");
}
function onVersionInfoNotify (data) {
    console.info("version is");
    console.info(data);
}

var global_openEye_SDK = null;
function initSDK(){
    global_openEye_SDK = new OpenEye_SDK({
            onOpeneyeReady: onOpeneyeReady,
            onOpeneyeClose: onOpeneyeClose,
            serviceStartUp: serviceStartUp,
            serviceShutDown: serviceShutDown,
            onOpeneyeLoginReady: onOpeneyeLoginReady,
            onOpeneyeLoginClose: onOpeneyeLoginClose,
            onOpeneyeCallReady: onOpeneyeCallReady,
            onOpeneyeCallClose: onOpeneyeCallClose,
            onVersionInfoNotify: onVersionInfoNotify
        });
}
```

5.1.2 config(配置)

接口描述

配置OpenEyeCall的运行参数。

注意事项

已经建立与OpenEyeCall的WebSocket连接。

方法定义

```
TUPCall.prototype.config = function(params, callbacks)
```


参数描述

表 5-3 参数说明

参数名	类型	可选/必选	描述
params	**Params**	必选	配置参数。
callbacks	**Callback**	可选	回调方法。

表 5-4 Params

参数名	类型	可选/必选	描述
log_path	String	可选	SIP消息日志存放路径。绝对路径或OpenEye安装目录的相对路径。 例如：“C:/log”，"./log"。 也可以使用"D:\\tup\\log"。 如果路径不存在，则会自动创建。 如果使用绝对路径，需确保各客户端均有指定的盘符，因而建议使用相对路径。
call	**Call**	必选	呼叫业务。
network	**Network**	必选	网络配置。
media	**Media**	必选	媒体设置。
audio	**Audio**	必选	音频设置。
account	**Account**	必选	账号密码类型设置。

表 5-5 Call

参数名	类型	可选/必选	描述
call_ipcall_enable	Number	必选	开启ip地址呼叫功能。 设置为0。

表 5-6 NetworkInfo

参数名	类型	可选/必选	描述
serverAddr	String	必选	SIP服务器IP地址
sipServerPort	Number	必选	SIP服务器端口，UDP默认5060，默认TLS端口5061。
sipTransportMode	Number	必选	sip信令传输模式，0为UDP，1为TLS。
httpPort	Number	必选	一般为0。

表 5-7 Sip

参数名	类型	可选/必选	描述
user_type	Number	必选	用户端类型。 设置为0。
tls_anonymous_enable	Number	必选	TLS匿名认证模式。0-不开启，1-开启 **说明** 匿名认证，存在安全风险，请谨慎开启。默认不开启。
tls_rootcertpath	String	可选	根证书完整路径。使用TLS传输时需要配置根证书。 例："F:/test/cert/root_cert_huawei.pem"
trans_mode	Number	必选	SIP传输协议。 ● 0：UDP(默认) ● 1：TLS ● 2：TCP

表 5-8 Media

参数名	类型	可选/必选	描述
trans_mode	Number	必选	媒体流加密模式。 设置为1 ，表示支持RTP（不加密）和SRTP（加密）。 **说明** RTP（不加密），存在安全风险，请谨慎使用。

表 5-9 Audio

参数名	类型	可选/必选	描述
audio_codec	String	必选	音频编解码优先级，以及支持的音频编解码方式。例如："112,98,18,9,8,0"。 ● 112：OPUS ● 98：iLBC ● 18：G729 ● 9：G722 ● 8：G711a, ● 0：G711u
dtmf_mode	Number	可选	DTMF (Dual Tone Multi Frequency) 模式，即按键声音和数据的传输模式。 ● 0：带内透传模式(默认) ● 1：RFC2833自动协商 ● 2：强制使用RFC2833协议 ● 4：info模式 ● 5：H245
audio_anr	Number	可选	噪音抑制。取值范围0-4,0表示关闭，1-4数值越大，噪音抑制强度越大，默认关闭。

参数名	类型	可选/必选	描述
audio_aec	Number	可选	回声消除，0关闭，1开启，默认关闭，建议开启。
audio_agc	Number	可选	自动增益，0关闭，1开启，默认关闭。

表 5-10 Account

参数名	类型	可选/必选	描述
account_pwd_type	Number	必选	账号密码类型。 设置为0。

表 5-11 Callback

参数名	类型	可选/必选	描述
callbacks	function	可选	回调方法。

表 5-12 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
local_ip	String	本地IP地址。 IPv4格式。如: "192.168.10.100"
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

使用示例

```
function sipBasicCfg() {
    global_cloudIPCC_SDK.tupCall.config({
        networkInfo: {
            serverAddr: "example.com",
            sipServerPort: 5060,
            sipTransportMode: "10.175.1.61",
            httpPort: 5060
```



```
        }
    },{response: configResponse});
}

function configResponse(data) {
    if (data.result == 0) {
        console.info("Config Success");
    } else {
        console.error("Config Failed");
        console.error(data);
    }
}
```

5.2 帐号注册与注销

5.2.1 register(注册)

接口描述

SIP帐号注册。

注意事项

- 已经建立与OpenEyeCall的WebSocket连接。
- 完成注册参数设置。

方法定义

```
OpenEyeCall.prototype.register = function(sip_num, sip_name, sip_pwd, sip_mode, callbacks)
```

参数描述

表 5-13 参数说明

参数名	类型	可选/必选	描述
sip_num	String	必选	用户号码。最大长度255个字符。 例："70942@example.com"
sip_name	String	必选	用户名，最大长度255个字符。
sip_pwd	String	必选	密码，最大长度255个字符（明文）。
sip_mode	Int	必选	组网模式,4:UAP组网;5:UAP pool组网
callbacks	**Callback**	必选	回调方法。

表 5-14 Callback

参数名	类型	可选/必选	描述
response	function	必选	注册结果的回调方法。回调方法的入参请参考表5-15。
onRegStatusUpdate	function	必选	回调方法的入参请参考表5-15上报注册状态。
onForceUnReg	function	必选	回调方法的入参请参考表5-15。

表 5-15 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调接口调用成功，并不表示注册成功，需要根据上报注册状态判断是否注册成功。

回调方法的入参示例：

```
{
  "description" : "tsdk_login",
  "result" : 0,
  "rsp" : 65537
}
```

使用示例

```
function register() {
    global_openEye_SDK.openEyeCall.register("70942@example.com", "70942@example.com", "1qaz@WSX",
4, {
        onRegStatusUpdate: onRegStatusUpdate,
        onForceUnReg: onForceUnRegInfo,
        response: registerResponse
    });
}

function onRegStatusUpdate(data){
    console.info(data);
}
function onForceUnReg(data){
    console.info(data);
}

function registerResponse(data) {
    if (data.result == 0) {
        console.info("Register Operation Success");
```



```
    } else {
        console.error("Register Operation Failed");
    }
}
```

5.2.2 deRegister(注销)

接口描述

SIP帐号注销。

注意事项

- 已经建立与OpenEyeCall的WebSocket连接。
- 对应的用户已注册。
- 未处于通话中。

方法定义

```
OpenEyeCall.prototype.deRegister = function(sip_num, callbacks)
```

参数描述

表 5-16 参数说明

参数名	类型	可选/必选	描述
sip_num	String	必选	用户号码.。最大长度255个字符。 例："70942@example.com"
callbacks	**Callback**	必选	回调方法。

表 5-17 Callback

参数名	类型	可选/必选	描述
response	function	必选	注册结果的回调方法。回调方法的入参请参考**表5-18**。

表 5-18 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_logout",
  "result" : 0,
  "rsp" : 65538
}
```

使用示例

```
function deRegister() {
    global_openEye_SDK.OpenEyeCall.deRegister("70942@example.com", {
        response: deRegisterResponse
    });
}
function deRegisterResponse(data) {
    if (data.result == 0) {
        console.info("DeRegister Success");
    } else {
        console.error("DeRegister Failed");
    }
}
```

5.2.3 事件

5.2.3.1 onRegStatusUpdate(上报注册状态)

事件描述

用户在发起注册之后，会通过该事件向用户上报两次注册状态，第一次通知当前帐号注册状态为：注册中。第二次通知当前帐号注册状态为：已注册或未注册。

事件示例

```
{
  "description" : "TSDK_E_LOGIN_EVT_LOGIN_SUCCESS",
  "notify" : 1004,
  "param" : {
    "loginSuccessInfo" : {"confEnvType":0},
    "serviceAccountType" : 4,
    "userId" : 0
  }
}
```


参数描述

表 5-19 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-20 Param

参数名	类型	描述
userId	Number	userid。
serviceAccountType	Number	账户类型取值： ● 4：UAP ● 5：UAP pool组网
loginSuccessInfo	Number	登录成功信息

5.3 音视频呼叫

5.3.1 setBasicCallEvent(设置基础呼叫事件)

接口描述

绑定呼叫相关事件的回调函数。

注意事项

已经完成注册。

方法定义

```
OpenEyeCall.prototype.setBasicCallEvent = function(callbacks)
```


参数描述

表 5-21 callbacks 参数说明

参数名	类型	可选/必选	描述
onCallIncoming	function	可选	回调方法的入参请参考**5.3.3.1 onCallIncoming(呼入事件)**。
onCallOutGoing	function	可选	回调方法的入参请参考**5.3.3.2 onCallOutGoing(呼出事件)**。
onCallRingBack	function	可选	回调方法的入参请参考**5.3.3.3 onCallRingBack(回铃事件)**。
onCallConnected	function	可选	回调方法的入参请参考**5.3.3.4 onCallConnected(呼叫接通事件)**。
onCallEnded	function	可选	回调方法的入参请参考**5.3.3.5 onCallEnded(呼叫结束事件)**。
onCallEndedFailed	function	可选	回调方法的入参请参考**5.3.3.6 onCallEndedFailed(呼叫结束失败事件)**。
onCallRtpCreated	function	可选	回调方法的入参请参考**5.3.3.7 onCallRtpCreated(RTP通道建立事件)**。

使用示例

```
function setBasicCallEvent(){
    global_openEye_SDK.openEyeCall.setBasicCallEvent({
        onCallIncoming: onCallIncoming,
        onCallOutGoing: onCallOutGoing,
        onCallRingBack: onCallRingBack,
        onCallConnected: onCallConnected,
        onCallEnded: onCallEnded,
        onCallEndedFailed: onCallEndedFailed,
        onCallRtpCreated: onCallRtpCreated
    });
}
function onCallIncoming(data){
    console.info(data);
}
function onCallOutGoing(data){
    console.info(data);
}
function onCallRingBack(data){
    console.info(data);
}
function onCallConnected(data){
    console.info(data);
}
function onCallEnded(data){
    console.info(data);
}
```



```
function onCallEndedFailed(data){
    console.info(data);
}
function onCallRtpCreated(data){
    console.info(data);
}
```

5.3.2 acceptCall(接听呼叫)

接口描述

收到呼入事件后，可使用该接口接听呼叫 。

注意事项

收到**呼入事件**。

方法定义

```
OpenEyeCall.prototype.acceptCall = function(callid, is_video_call, callbacks)
```

参数描述

表 5-22 参数说明

参数名	类型	可选/必选	描述
callid	Number	必选	呼叫ID，由上**onCallIncoming**上报。
is_video_call	Number	必选	是否为视频呼叫，由**onCallIncoming**上报。
callbacks	**Callback**	必选	回调方法。

表 5-23 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表5-24**

表 5-24 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_accept_call",
  "result" : 0,
  "rsp" : 67538
}
```

使用示例

```
function acceptCall() {
    global_openEye_SDK.openEyeCall.acceptCall(1917517824, 0, { response: onAcceptCallReponse });
}

function onAcceptCallReponse(data) {
    if (data.result == 0) {
        console.error("AcceptCall success. ");
    } else {
        console.error("AcceptCall failed. The ErrorCode is " + data.result);
        console.info(data);
    }
}
```

5.3.3 事件

5.3.3.1 onCallIncoming(呼入事件)

事件描述

对方发起呼叫时，本端收到来电事件，参数中携带呼叫ID以及主叫号码。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_CALL_INCOMING",
  "notify" : 2002,
  "param":
   {
   "callId":1559166976,
   "callInfo":
      {
      "callId":1559166976,
      "callState":1,
      "confId":"",
      "confPasscode":"",
      "isAutoAnswer":0,
      "isCaller":0,
      "isFocus":0,
      "isVideoCall":0,
      "peerDisplayName":"",
```



```
        "peerNumber":"444002",
        "reasonCode":0,
        "reasonDescription":"",
        "sipAccountID":0
        },
    "maybeVideoCall":0
    }
}
```

参数描述

表 5-25 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-26 Param

参数名	类型	描述
callId	Number	呼叫ID。
peerNumber	String	对端电话号码。

说明

Param中主要关注callId和peerNumber字段。

5.3.3.2 onCallOutGoing(呼出事件)

事件描述

本方发起呼叫时，本端收到呼出事件，参数中携带呼叫ID。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_CALL_OUTGOING",
  "notify" : 2003,
  "param":
   {
   "callId":1867907072,
   "callInfo":
      {
      "callId":1867907072,
      "callState":2,
      "confId":"",
      "confPasscode":"",
      "isAutoAnswer":0,
      "isCaller":1,
      "isFocus":0,
```



```
        "isVideoCall":0,
        "peerDisplayName":"",
        "peerNumber":"444002",
        "reasonCode":0,
        "reasonDescription":"",
        "sipAccountID":0
        }
    }
}
```


参数描述

表 5-27 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-28 Param

参数名	类型	描述
callId	Number	呼叫ID。
peerNumber	String	对端电话号码。

说明

Param中主要关注callId和peerNumber字段。

5.3.3.3 onCallRingBack(回铃事件)

事件描述

本方发起呼叫后，本端收到回铃事件，参数中携带呼叫ID。

事件示例


```
{
  "description" : "TSDK_E_CALL_EVT_CALL_RINGBACK",
  "notify" : 2004,
  "param" : {
    "callId" : 1867907072
  }
}
```


参数描述

表 5-29 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-30 Param

参数名	类型	描述
callId	Number	呼叫ID。

说明

Param中主要关注callId字段。

5.3.3.4 onCallConnected(呼叫接通事件)

事件描述

呼叫接通。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_CALL_CONNECTED",
  "notify" : 2006,
  "param":
   {
   "callId":1559166976,
   "callInfo":
      {
      "callId":1559166976,
      "callState":3,
      "confId":"",
      "confPasscode":"",
      "isAutoAnswer":0,
      "isCaller":0,
      "isFocus":0,
      "isVideoCall":0,
      "peerDisplayName":"",
      "peerNumber":"444002",
      "reasonCode":0,
      "reasonDescription":"",
      "sipAccountID":0
      }
   }
}
```


参数描述

表 5-31 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-32 Param

参数名	类型	描述
callId	Number	呼叫ID。
peerNumber	String	对端电话号码。

说明

Param中主要关注callId和peerNumber字段。

5.3.3.5 onCallEnded(呼叫结束事件)

事件描述

呼叫建立过程中或呼叫建立后，一方挂断呼叫，呼叫结束，触发该事件。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_CALL_ENDED",
  "notify" : 2007,
  "param":
   {
   "callId":1559166976,
   "callInfo":
      {
      "callId":1559166976,
      "callState":5,
      "confId":"",
      "confPasscode":"",
      "isAutoAnswer":0,
      "isCaller":0,
      "isFocus":0,
      "isVideoCall":0,
      "peerDisplayName":"",
      "peerNumber":"444002",
      "reasonCode":0,
      "reasonDescription":"",
      "sipAccountID":0
      }
   }
}
```


参数描述

表 5-33 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-34 Param

参数名	类型	描述
callId	Number	呼叫ID。
peerNumber	String	对端电话号码。

说明

Param中主要关注callId和peerNumber字段。

5.3.3.6 onCallEndedFailed(呼叫结束失败事件)

事件描述

结束呼叫时，如果传入的callid对应的呼叫不存在，则返回呼叫结束失败事件。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_ENDCALL_FAILED",
  "notify" : 2022,
  "param" : {
    "callId" : 0,
    "result" : 50331670
  }
}
```

参数描述

表 5-35 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-36 Param

参数名	类型	描述
callId	Number	呼叫ID。
result	Number	失败原因码。

5.3.3.7 onCallRtpCreated(RTP 通道建立事件)

事件描述

通话接通后，媒体通道建立，触发该事件。

事件示例

```
{
  "description" : "TSDK_E_CALL_EVT_CALL_RTP_CREATED",
  "notify" : 2005,
  "param" : {
    "callId" : 1867907072
  }
}
```

参数描述

表 5-37 参数说明

参数名	类型	描述
description	String	当前请求描述。
notify	Number	内部事件编号。
param	**Param**	事件内容。

表 5-38 Param

参数名	类型	描述
callId	Number	呼叫ID。

6 音视频呼叫接口扩展

6.1 音视频呼叫

6.1.1 startCall(发起呼叫)

接口描述

发起一路VOIP呼叫。

注意事项

- 已经完成账号注册。
- 已经设置基础呼叫事件。

方法定义

```
OpenEyeCall.prototype.startCall = function(callee_num, is_video_call, callbacks)
```

参数描述

表 6-1 参数说明

参数名	类型	可选/必选	描述
callee_num	String	必选	被叫号码，最大长度255个字符。

参数名	类型	可选/必选	描述
is_video_call	Boolean	必选	呼叫类型。填写false。
callbacks	**Callback**	必选	回调方法。

表 6-2 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-3**

表 6-3 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	呼叫信息。

表 6-4 Param 参数说明

参数名	类型	描述
call_id	Number	呼叫ID，OpenEye会自动填充在回调函数的参数中。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_start_call",
  "param" : {
    "callId" : 1541472256
  },
  "result" : 0,
  "rsp" : 67537
}
```

使用示例

```
var tupCurrentCallId;
function startCall() {
```



```
    global_openEye_SDK.openEyeCall.openEyeCall("70943", false, {
        response: startCallResponse
    });
}

function startCallResponse(data) {
    if (data.result == 0) {
        console.info("StartCall success. ");
        tupCurrentCallId = data.param.call_id;
    } else {
        console.error("StartCall failed. The ErrorCode is " + data.result);
        console.info(data);
    }
}
```

6.1.2 endCall(结束呼叫)

接口描述

结束和其他用户的通话或者来电。

注意事项

有其他用户的通话或者来电。

方法定义

```
OpenEyeCall.prototype.endCall = function(callid, callbacks)
```

参数描述

表 6-5 参数说明

参数名	类型	可选/必选	描述
callid	Number	必选	呼叫ID。
callbacks	**Callback**	必选	回调方法。

表 6-6 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-7**

表 6-7 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_end_call",
  "result" : 0,
  "rsp" : 67539
}
```

使用示例

```
function endCall() {
  global_openEye_SDK.openEyeCall.endCall(1755709440, { response: onEndCallReponse });
}

function onEndCallReponse(data) {
    if (data.result == 0) {
        console.error("EndCall success. ");
    } else {
        console.error("EndCall failed. The ErrorCode is " + data.result);
        console.info(data);
    }
}
```

6.1.3 operateMic(闭音麦克风)

接口描述

设置（或取消）麦克风静音。

注意事项

已经建立通话。

方法定义

```
OpenEyeCall.prototype.operateMic = function(callid, to_mute, callbacks)
```

参数描述

表 6-8 参数说明

参数名	类型	可选/必选	描述
callid	Number	必选	呼叫ID。

参数名	类型	可选/必选	描述
to_mute	Number	必选	是否静音。1表示静音，0表示恢复通话。
callbacks	**Callback**	可选	回调方法。

表 6-9 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-10**

表 6-10 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_mute_mic",
  "result" : 0,
  "rsp" : 67547
}
```

使用示例

```
/**
 * 静音/取消静音
 * @param {*} flag 1表示静音，0表示取消静音
 */
function operateMic(flag) {
    global_openEye_SDK.openEyeCall.operateMic(1755709440, flag, { response: onOperateMicResponse });
}

function onOperateMicResponse(data) {
    if (data.result == 0) {
        console.info("OperateMic success. ");
    } else {
        console.error("OperateMic failed. The ErrorCode is " + data.result);
        console.info(data);
    }
}
```


6.1.4 dtmf(二次拨号)

接口描述

在通话中发送二次拨号信息 。

注意事项

处于通话中才可以发送二次拨号信息。

方法定义

```
OpenEyeCall.prototype.dtmf = function(callid, keyTone, callbacks)
```

参数描述

表 6-11 参数说明

参数名	类型	可选/必选	描述
callid	Number	必选	呼叫ID。
keyTone	Number	必选	DTMF键值。取值范围0-16。
callbacks	**Callback**	可选	回调方法。

表 6-12 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-13**

表 6-13 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_send_dtmf",
  "result" : 0,
  "rsp" : 67540
}
```

使用示例

```
function dtmf() {
    global_openEye_SDK.openEyeCall.dtmf(1755709440, 12, {
        response: dtmfResponse
    });
}

function dtmfResponse(data) {
    if (data.result == 0) {
        console.info("Dtmf success. ");
    } else {
        console.error("Dtmf failed. The ErrorCode is " + data.result);
        console.info(data);
    }
}
```

6.1.5 screenShot(截屏)

接口描述

截取对端视频的一帧图像并保存。

注意事项

前置条件：已经建立与OpenEyeCall的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.screenShot = function(callbacks)
```

参数描述

表 6-14 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	必选	回调方法。

表 6-15 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考表6-16

表 6-16 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_stopsharewindow",
  "result" : 0,
  "rsp" : 67762
}
```

使用示例

```
function startScreenShot(){
 console.info("startScreenShot");
 this.global_openEye_SDK.openEyeCall.screenShot({ response: startScreenShotResponse })
}

function startScreenShotResponse(data){
 console.log(data);
 if (data.result == 0) {
        console.info("startScreenShot Success");
    } else {
        console.error("startScreenShot failed");
        console.error(data);
    }
}
```

6.1.6 catchVideo(录屏)

接口描述

开始或者结束录屏。

注意事项

前置条件：已经建立与OpenEyeCall的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.videoCatch = function(value, callbacks)
```

参数描述

表 6-17 参数说明

参数名	类型	可选/必选	描述
value	bool	必选	true为启动录屏，false为停止录屏。
callbacks	**Callback**	必选	回调方法。

表 6-18 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-19**

表 6-19 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_stopsharewindow",
  "result" : 0,
  "rsp" : 67763
}
```

使用示例

```
function catchVideo(value){
 this.global_openEye_SDK.openEyeCall.videoCatch(value, { response: startVideoCatchResponse })
}

function startVideoCatchResponse(data){
 console.log(data);
```



```
if (data.result == 0) {
        console.info("startVideoCatch Success");
    } else {
        console.error("startVideoCatch failed");
        console.error(data);
    }
}
```

6.1.7 setAnswerWay(设置接听方式)

接口描述

设置接听方式（自动挂断、自动接听、免打扰）。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.setAnswerWay = function(answerWay, times, callbacks)
```

参数描述

表 6-20 参数说明

参数名	类型	可选/必选	描述
answerWay	string	必选	1:自动挂断，2:自动接听，4:免打扰
times	Number	必选	对应接听方式作用的时间间隔（免打扰设置无效，可选）
callbacks	**Callback**	必选	回调方法。

表 6-21 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-22**

表 6-22 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
errMsg	String	错误信息

回调方法的入参示例：

```
{
    description: "SetAnswerWay",
    result: 0,
    rsp: 67767,
    errMsg: ""
}
```

使用示例

```
function setAnswerWay(answerWay, times){
 this.global_openEye_SDK.openEyeCall.setAnswerWay(answerWay, times, { response:
onSetAnswerWayResponse });
}

function onSetAnswerWayResponse(data){
 console.log(data);
 if (data.result == 0) {
        console.info("SetAnswerWay Success");
    } else {
        console.error("SetAnswerWay failed. The ErrorCode is " + data.result + ";errMsg:" + data.errMsg);
        alert("SetAnswerWay failed.  The ErrorCode is " + data.result + ";errMsg:" + data.errMsg);
    }
}
```

6.2 设备管理

6.2.1 getMediaDevices(获取设备列表)

接口描述

获取本地设备列表，如麦克风和扬声器。

注意事项

需要账号注册成功后才能调用该接口。

方法定义

```
OpenEyeCall.prototype.getMediaDevices = function(type, callbacks)
```


参数描述

表 6-23 参数说明

参数名	类型	可选/必选	描述
type	Number	必选	设备类型。0表示麦克风，1表示扬声器。
callbacks	**Callback**	可选	回调方法。

表 6-24 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-25**

表 6-25 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	设备信息。

表 6-26 Param

参数名	类型	描述
array	**设备**的数组	设备列表。
device_num	Number	设备数。

表 6-27 设备

参数名	类型	描述
camera_orient	Number	预留字段
index	Number	设备序号。

参数名	类型	描述
name	String	设备名称。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_get_devices",
  "param" : {
    "deviceInfo" : [
      {
            "cameraOrient":0,
      "deviceId":0,
      "deviceName":
      "default: 扬声器 (Huawei HDP Audio Driver)",
      "index":0

      },
      {
            "cameraOrient":0,
      "deviceId":0,
      "deviceName":"扬声器 (Huawei HDP Audio Driver)",
      "index":1
      }
    ],
    "deviceType":1,
    "num" : 2
  },
  "result" : 0,
  "rsp" : 67550
}
```

使用示例

```
function getMediaDevices() {
    global_openEye_SDK.openEyeCall.getMediaDevices(1, {
        response: getMediaDevicesResponse
    });
}

function getMediaDevicesResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("GetMediaDevices success");
    } else {
        console.error("GetMediaDevices failed");
    }
}
```

6.2.2 setMicIndex(设置麦克风)

接口描述

设置用于通话的麦克风。如果不设置，OpenEye将使用系统默认麦克风。

注意事项

- 已经建立与OpenEye的WebSocket连接。
- 设备序号一般在系统初始化后通过**getMediaDevice**获取。

方法定义

```
OpenEyeCall.prototype.setMicIndex = function(idx, callbacks)
```

参数描述

表 6-28 参数说明

参数名	类型	可选/必选	描述
idx	Number	必选	麦克风设备序号。
callbacks	**Callback**	可选	回调方法。

表 6-29 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-30**

表 6-30 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_mic_index",
  "result" : 0,
  "rsp" : 67551
}
```

使用示例

```
function setMicIndex() {
    global_openEye_SDK.openEyeCall.setMicIndex(1, {
        response: setMicIndexResponse
    });
}

function setMicIndexResponse(data) {
    console.info(data);
```



```
    if (data.result == 0) {
        console.info("SetMicIndex success")
    } else {
        console.error("StMicIndex failed");
    }
}
```

6.2.3 mediaGetMicIndex(查询当前使用的麦克风)

接口描述

查询当前使用的麦克风，返回麦克风对应设备序号。

注意事项

- 已经建立与OpenEye的WebSocket连接。
- 用于接口测试或产品调试，实际产品业务场景中无需调用。

方法定义

```
OpenEyeCall.prototype.mediaGetMicIndex = function(callbacks)
```

参数描述

表 6-31 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	可选	回调方法。

表 6-32 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-33**

表 6-33 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	当前使用的麦克风。

表 6-34 Param

参数名	类型	描述
index	Number	当前麦克风设备序号。

说明

回调方法的入参示例：

```
{
   "description" : "tsdk_get_mic_index",
   "param" : {
      "index" : 0
   },
   "result" : 0,
   "rsp" : 67552
}
```

使用示例

```
function mediaGetMicIndex() {
    global_openEye_SDK.openEyeCall.mediaGetMicIndex({ response: mediaGetMicIndexResponse });
}

function mediaGetMicIndexResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("MediaGetMicIndex success")
    } else {
        console.error("MediaGetMicIndex failed");
    }
}
```

6.2.4 setSpeakIndex (设置扬声器)

接口描述

设置用于通话的扬声器。

注意事项

- 已经建立与OpenEye的WebSocket连接。
- 设备序号一般在系统初始化后通过**getMediaDevice**获取。

方法定义

```
OpenEyeCall.prototype.setSpeakIndex = function(idx, callbacks)
```


参数描述

表 6-35 参数说明

参数名	类型	可选/必选	描述
idx	Number	必选	扬声器设备序号。
callbacks	**Callback**	可选	回调方法。

表 6-36 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-37**

表 6-37 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_speak_index",
  "result" : 0,
  "rsp" : 67553
}
```

使用示例

```
function setSpeakIndex() {
    global_openEye_SDK.openEyeCall.setSpeakIndex(1, {
        response: setSpeakIndexResponse
    });
}

function setSpeakIndexResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("SetSpeakIndex success")
    } else {
        console.error("SetSpeakIndex failed");
```



```
    }
}
```

6.2.5 mediaGetSpeakIndex (查询当前使用的扬声器)

接口描述

查询当前使用的扬声器，接口返回扬声器设备序号。

注意事项

- 已经建立与OpenEye的WebSocket连接。
- 用于接口测试或产品调试，实际产品业务场景中无需调用。

方法定义

```
OpenEyeCall.prototype.mediaGetSpeakIndex = function(callbacks)
```

参数描述

表 6-38 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	可选	回调方法。

表 6-39 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-40**

表 6-40 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	当前使用的麦克风。

表 6-41 Param

参数名	类型	描述
index	Number	当前麦克风设备序号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_get_speak_index",
  "param" : {
    "index" : 0
  },
  "result" : 0,
  "rsp" : 67554
}
```

使用示例

```
function mediaGetSpeakIndex() {
    global_openEye_SDK.openEyeCall.mediaGetSpeakIndex({ response: mediaGetSpeakIndexResponse });
}

function mediaGetSpeakIndexResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("MediaGetSpeakIndex success");
    } else {
        console.error("MediaGetSpeakIndex failed");
    }
}
```

6.2.6 setMicVol(设置麦克风音量)

接口描述

设置麦克风音量大小。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.setMicVol = function(volume, device, callbacks)
```

参数描述

表 6-42 参数说明

参数名	类型	可选/必选	描述
volume	Number	必选	音量大小，取值范围[0, 100]。

参数名	类型	可选/必选	描述
device	Number	必选	设备类型。设置为1。
callbacks	**Callback**	可选	回调方法。

表 6-43 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-44**

表 6-44 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_mic_volume",
  "result" : 0,
  "rsp" : 67577
}
```

使用示例

```
function setMicVol() {
    global_openEye_SDK.openEyeCall.setMicVol(20, 1, {
        response: setMicVolReponse
    });
}

function setMicVolReponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("SetMicVol Success.")
    } else {
        console.error("SetMicVol failed.");
    }
}
```


6.2.7 getMicVol(查询麦克风当前音量)

接口描述

查询麦克风当前音量大小。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.getMicVol = function(callbacks)
```

参数描述

表 6-45 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	可选	回调方法。

表 6-46 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-47**

表 6-47 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	音量信息。

表 6-48 Param

参数名	类型	描述
volume	Number	当前麦克风音量大小。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_get_mic_volume",
  "param" : {
    "volume" : 20
  },
  "result" : 0,
  "rsp" : 67578
}
```

使用示例

```
function getMicVol() {
    global_openEye_SDK.openEyeCall.getMicVol({
        response: getMicVolResponse
    });
}

function getMicVolResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("GetMicVol Success.");
    } else {
        console.error("GetMicVol failed.");
    }
}
```

6.2.8 setSpkVol(设置扬声器音量)

接口描述

设置扬声器音量大小。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.setSpkVol = function(volume, device, callbacks)
```


参数描述

表 6-49 参数说明

参数名	类型	可选/必选	描述
volume	Number	必选	音量大小，取值范围[0,100]。
device	Number	必选	设备类型。设置为0。
callbacks	**Callback**	可选	回调方法。

表 6-50 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-51**

表 6-51 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_speak_volume",
  "result" : 0,
  "rsp" : 67557
}
```

使用示例

```
function setSpkVol() {
    global_openEye_SDK.openEyeCall.setSpkVol(80, 0, {
        response: setSpkVolResponse
    });
}

function setSpkVolResponse(data) {
    console.info(data);
    if (data.result == 0) {
```



```
        console.info("SetSpkVol Success.")
    } else {
        console.error("SetSpkVol failed.");
    }
}
```

6.2.9 getSpkVol(查询扬声器当前音量)

接口描述

查询扬声器当前音量大小。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.getSpkVol = function(callbacks)
```

参数描述

表 6-52 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	可选	回调方法。

表 6-53 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-54**

表 6-54 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	查询结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	**Param**	音量信息。

表 6-55 Param

参数名	类型	描述
volume	Number	当前输出音量大小。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_get_speak_volume",
  "param" : {
    "volume" : 80
  },
  "result" : 0,
  "rsp" : 67558
}
```

使用示例

```
function getSpkVol() {
    global_openEye_SDK.openEyeCall.getSpkVol({
        response: getSpkVolResponse
    });
}

function getSpkVolResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("GetSpkVol Success.");
    } else {
        console.error("GetSpkVol failed");
    }
}
```

6.2.10 setVideoWindowParam(设置视频窗口位置和宽高)

接口描述

设置视频通话时的视频画面参数，包括位置，宽高（单位为像素）。

注意事项

已经建立与OpenEye的WebSocket连接。

注意视频画面的推荐宽高为720px*480px。最小可显示操作UI的宽高为480px*360px。低于480px*360px则仅显示视频画面，不显示操作控制UI（不建议设低于该尺寸的数值）。

视频通话接听之前必须首先调用该接口预先设置好视频窗口信息，一旦调用该接口设置视频窗口后长期有效，直到再次调用该接口修改或者整个页面关闭才会改变。

方法定义

```
OpenEyeCall.prototype.setVideoWindowParam= function(posX,posY,width,height,callbacks)
```


参数描述

表 6-56 参数说明

参数名	类型	可选/必选	描述
posX	Number	必选	视频窗口左上角x坐标，大于0。
posY	Number	必选	视频窗口左上角y坐标，大于0。
width	Number	必选	视频窗口宽度
height	Number	必选	视频窗口高度
callbacks	**Callback**	可选	回调方法。

表 6-57 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-58**

表 6-58 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_video_rect",
  "result" : 0,
  "rsp" : 67745
}
```

使用示例

```
function setVideoWindowParam() {
this.global_openEye_SDK.openEyeCall.setVideoWindowParam(20,30,720,480, { response:
setVideoWindowParamResponse })
```



```
}

function setVideoWindowParamResponse(data) {
    if (data.result == 0) {
        console.info("setVideoWindowParam Success");
    } else {
        console.error("setVideoWindowParam failed");
    }
}
```

6.2.11 setVideoLayoutMode(设置视频窗口画面排列模式)

接口描述

设置视频通话时的视频布局模式，并列，画中画模式两种。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.setVideoLayoutMode = function(layoutMode,callbacks)
```

参数描述

表 6-59 参数说明

参数名	类型	可选/必选	描述
layoutMode	Number	必选	视频模式，0,画中画模式。1,并列模式。
callbacks	**Callback**	可选	回调方法。

表 6-60 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-61**

表 6-61 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。

参数名	类型	描述
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_video_layout_mode",
  "result" : 0,
  "rsp" : 67749
}
```

使用示例

```
function setVideoWindowParam() {
this.global_openEye_SDK.openEyeCall.setVideoLayoutMode(param, { response:
setVideoLayoutModeResponse })
}
function setVideoLayoutModeResponse(data) {
    if (data.result == 0) {
        console.info("setVideoLayoutMode Success");
    } else {
        console.error("setVideoLayoutMode failed");
    }
}
```

6.2.12 setVideoDisplayMode(设置视频窗口画面裁剪模式)

接口描述

设置视频通话时的视频画面裁剪参数。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.setVideoDisplayMode = function(displayMode,callbacks)
```

参数描述

表 6-62 参数说明

参数名	类型	可选/必选	描述
displayMode	Number	必选	视频画面裁剪模式，1,(不拉伸)黑边模式。2,(不拉伸)裁剪模式。
callbacks	**Callback**	可选	回调方法。

表 6-63 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考表6-64

表 6-64 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_set_video_display_mode",
  "result" : 0,
  "rsp" : 67751
}
```

使用示例

```
function setVideoWindowParam() {
this.global_openEye_SDK.openEyeCall.setVideoDisplayMode(param, { response:
setVideoDisplayModeResponse })
}
function setVideoDisplayModeResponse(data) {
    if (data.result == 0) {
        console.info("setVideoDisplayMode Success");
    } else {
        console.error("setVideoDisplayMode failed");
    }
}
```

6.2.13 openCamera(打开摄像头)

接口描述

打开本端摄像头。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.openCamera = function(callId, callbacks)
```

参数描述

表 6-65 参数说明

参数名	类型	可选/必选	描述
callId	Number	必选	当前通话的callID，在没有进行中的通话时则须设置为-1。
callbacks	**Callback**	可选	回调方法。

表 6-66 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-67**

表 6-67 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_control_camera",
  "result" : 0,
  "rsp" : 67759
}
```

使用示例

```
function switchCameraMode() {
 var ischecked = document.getElementById("camera-control-toggle-button").checked;
 if (tupCurrentCallId == "") {
  tupCurrentCallId = -1;
 }
 if (ischecked) {
  console.info("switchCameraMode ischecked true.CallId is:"+tupCurrentCallId);
  this.global_openEye_SDK.openEyeCall.openCamera(tupCurrentCallId, {
   response: cameraModeResponse});
 } else {
```



```
 console.info("switchCameraMode ischecked false.CallId is:"+tupCurrentCallId);
 this.global_openEye_SDK.openEyeCall.closeCamera(tupCurrentCallId, {
  response: cameraModeResponse});
 }
}

function cameraModeResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("controlVideo Success.");
    } else {
        console.error("controlVideo failed.");
    }
}
```

6.2.14 closeCamera(关闭摄像头)

接口描述

关闭本端摄像头。

注意事项

已经建立与OpenEye的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.closeCamera = function(callId, callbacks)
```

参数描述

表 6-68 参数说明

参数名	类型	可选/必选	描述
callId	Number	必选	当前通话的callID，在没有进行中的通话时则须设置为-1。
callbacks	**Callback**	可选	回调方法。

表 6-69 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-70**

表 6-70 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
   "description" : "tsdk_control_camera",
   "result" : 0,
   "rsp" : 67759
}
```

使用示例

```
function switchCameraMode() {
 var ischecked = document.getElementById("camera-control-toggle-button").checked;
 if (tupCurrentCallId == "") {
  tupCurrentCallId = -1;
 }
 if (ischecked) {
  console.info("switchCameraMode ischecked true.CallId is:"+tupCurrentCallId);
  this.global_openEye_SDK.openEyeCall.openCamera(tupCurrentCallId, {
   response: cameraModeResponse});
 } else {
  console.info("switchCameraMode ischecked false.CallId is:"+tupCurrentCallId);
  this.global_openEye_SDK.openEyeCall.closeCamera(tupCurrentCallId, {
   response: cameraModeResponse});
 }
}

function cameraModeResponse(data) {
    console.info(data);
    if (data.result == 0) {
        console.info("controlVideo Success.");
    } else {
        console.error("controlVideo failed.");
    }
}
```

6.3 屏幕共享

共享本端屏幕给对方，屏幕共享功能包括桌面共享，指定区域共享，指定程序窗口共享三个功能。注意，屏幕共享功能的前提条件是处于视频通话状态中，音频通话不支持此功能。

6.3.1 获取可共享程序列表

接口描述

获取当前操作系统可以被共享的程序窗口列表。

注意事项

前置条件:已经建立与OpenEye的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.getAppList = function(callbacks)
```

参数描述

表 6-71 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	必选	回调方法。

表 6-72 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-73**

表 6-73 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。
param	key:value	窗口句柄与窗口名称的key value键值对

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_getapplist",
  "result" : 0,
  "rsp" : 67753
  "param":{
    65552: "桌面"
    132070: "app1"
    132974: "app2"
    198240: "app3"
    328180: "app4"
    329712: "app5"
  }
}
```

注意:65552这些数字为对应后面窗口的window系统内的窗口句柄，调用**设置要共享窗口的接口**入参的第一个参数即为该值

使用示例

```
function getAppList(){
    this.global_openEye_SDK.openEyeCall.getAppList({ response: getAppListResponse })
}
function getAppListResponse(data) {
    console.log(data);
    if (data.result == 0) {
        console.info("getAppListResponse success");
        document.getElementById("shareAppList").innerHTML = "";
        for (var key in data.param) {
            var item = data.param[key];
            document.getElementById("shareAppList").options.add(new Option(key + "_" + item, key));
        }
    } else {
        console.error("getAppListResponse failed");
    }
}
```

6.3.2 设置要共享的程序信息

接口描述

设置将要被共享窗口信息。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.setShareWindow = function(hwnd,callbacks)
```


参数描述

表 6-74 参数说明

参数名	类型	可选/必选	描述
hwnd	int	必选	要共享的窗口句柄，详情见**6.3.1 获取可共享程序列表**中的接口说明。
callbacks	**Callback**	必选	回调方法。

表 6-75 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-76**

表 6-76 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_setshareapp",
  "result" : 0,
  "rsp" : 67754
}
```

使用示例

```
function setShareApp(){
    this.global_openEye_SDK.openEyeCall.setShareWindow(document.getElementById("shareAppList").value,
{ response: setShareAppRespone })
}
//step2 callback,设置要共享的窗口的callback
function setShareAppRespone(data){
    console.log(data);
    if (data.result == 0) {
```



```
        console.info("setShareApp Success");
    } else {
        console.error("setShareApp failed");
    }
}
```

6.3.3 开始共享

接口描述

发起方主动发起窗口共享。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.startShareWindow = function(callbacks)
```

参数描述

表 6-77 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	必选	回调方法。

表 6-78 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-79**

表 6-79 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_startsharewindow",
  "result" : 0,
  "rsp" : 67755
}
```

使用示例

```
function startShare(){
    this.global_openEye_SDK.openEyeCall.startShareWindow({ response: startShareRespone })
}
function startShareRespone(data){
    console.log(data);
    if (data.result == 0) {
        console.info("startShare Success");
    } else {
        console.error("startShare failed");
    }
}
```

6.3.4 结束共享

接口描述

发起方主动结束共享。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接，且处于视频通话中。

方法定义

```
OpenEyeCall.prototype.stopShareWindow = function(callbacks)
```

参数描述

表 6-80 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	必选	回调方法。

表 6-81 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-82**

表 6-82 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_stopsharewindow",
  "result" : 0,
  "rsp" : 67756
}
```

使用示例

```
function stopShare(){
    this.global_openEye_SDK.openEyeCall.stopShareWindow({ response: stopShareRespone })
}
function stopShareRespone(data){
    console.log(data);
    if (data.result == 0) {
        console.info("stopShare Success");
    } else {
        console.error("stopShare failed");
    }
}
```

6.3.5 共享开关

接口描述

控制共享功能是否启用。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接。

方法定义

```
OpenEyeCall.prototype.shareControl = function(value, callbacks)
```

参数描述

表 6-83 参数说明

参数名	类型	可选/必选	描述
value	bool	必选	共享开关是否打开。
callbacks	**Callback**	必选	回调方法。

表 6-84 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法的入参请参考**表6-85**

表 6-85 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
result	Number	设置结果。0表示成功，其他表示失败。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "tsdk_share_evt_stopsharewindow",
  "result" : 0,
  "rsp" : 67760
}
```

使用示例

```
function switchShare() {
 var ischecked = document.getElementById("share-control-toggle-button").checked;
 if (ischecked) {
  console.info("switchShare ischecked true.");
  shareSwitch = true;
  document.getElementById("shareControlDiv").style.visibility = "visible";
  this.global_openEye_SDK.openEyeCall.shareControl(shareSwitch, { response: switchShareRespone })
 } else {
  console.info("switchShare ischecked false.");
  shareSwitch = false;
  document.getElementById("shareControlDiv").style.visibility = "hidden";
  this.global_openEye_SDK.openEyeCall.shareControl(shareSwitch, { response: switchShareRespone })
```



```
 }
}

function switchShareRespone(data){
 console.log(data);
 if (data.result == 0) {
        console.info("switchShare Success");
    } else {
        console.error("switchShare failed");
        console.error(data);
    }
}
```

6.4 截屏

6.4.1 截图返回路径

接口描述

视频通话过程中截取一张对方的视频画面并返回图片路径。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接并处于视频通话中。

方法定义

```
OpenEyeCall.prototype.screenShot = function(callbacks)
```

参数描述

表 6-86 参数说明

参数名	类型	可选/必选	描述
callbacks	**Callback**	必选	回调方法。

表 6-87 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法。

表 6-88 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
path	String	若成功，则是生成图片的保存路径，若失败，则无此字段
result	Number	设置结果。1表示失败，若成功则无此字段。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "OEScreenShot",
  "result" : 1,
  "rsp" : 67762
}

{
  "description" : "OEScreenShot",
  "path" : "xxx",
  "rsp" : 67762
}
```

使用示例

```
function startScreenShot(){
    console.info("startScreenShot");
    this.global_openEye_SDK.openEyeCall.screenShot({ response: startScreenShotResponse })
}

function startScreenShotResponse(data){
    console.log(data);
    if (data.result == 0) {
        console.info("startScreenShot Success");
    } else {
        console.error("startScreenShot failed");
        console.error(data);
    }
}
```

6.4.2 截图返回 base64 编码

接口描述

视频通话过程中截取一张对方的视频画面并返回图片base64编码。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接并处于视频通话中。

方法定义

```
OpenEyeCall.prototype.screenShotBase64 = function(callbacks)
```


参数描述

表 6-89 参数说明

参数名	类型	可选/必选	描述
callbacks	Callback	必选	回调方法。

表 6-90 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法。

表 6-91 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
base64	String	若成功，则是图片base64编码，若失败，则无此字段
result	Number	设置结果。1表示失败，若成功则无此字段。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "OEScreenShot",
  "result" : 1,
  "rsp" : 67762
}

{
  "description" : "OEScreenShot",
  "base64" : "xxx",
  "rsp" : 67762
}
```

使用示例

```
function startScreenShotBase64(){
    console.info("startScreenShot");
    this.global_openEye_SDK.openEyeCall.screenShotBase64({ response: startScreenShotResponse })
}

function startScreenShotResponse(data){
```



```
    console.log(data);
    if (data.result == 0) {
        console.info("startScreenShot Success");
    } else {
        console.error("startScreenShot failed");
        console.error(data);
    }
}
```

6.5 录屏

6.5.1 开始录屏与结束录屏

接口描述

视频通话过程中开始或结束录制一段对方画面的视频，返回视频路径。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接并处于视频通话中。

方法定义

```
OpenEyeCall.prototype.videoCatch = function(operation, callbacks)
```

参数描述

表 6-92 参数说明

参数名	类型	可选/必选	描述
operation	int	必选	进行的操作类型，0为开始录屏，1为停止录屏
callbacks	**Callback**	必选	回调方法。

表 6-93 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法。

表 6-94 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。

参数名	类型	描述
path	String	若成功，则是视频的路径，结束录屏成功时返回
result	Number	设置结果。0表示成功，1表示失败，若无此字段表示操作成功。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "OECatchVideo",
  "result" : 1,
  "rsp" : 67762
}

{
  "description" : "OECatchVideo",
  "path" : "xxx",
  "rsp" : 67762
}
```

使用示例

```
function catchVideo(operation){
    this.global_openEye_SDK.openEyeCall.videoCatch(operation, { response: startVideoCatchResponse })
}

function startVideoCatchResponse(data){
    console.log(data);
    if (data.result == 0) {
        console.info("VideoCatch Success");
    } else {
        console.error("VideoCatch failed");
        console.error(data);
    }
}
```

6.5.2 开始录屏与结束录屏(定时返回 base64 编码)

接口描述

视频通话过程中开始或结束录制一段对方画面的视频，并返回base64编码，可实现定时录制。

注意事项

前置条件:已经建立与OpenEyeCall的WebSocket连接并处于视频通话中。

方法定义

```
OpenEyeCall.prototype.videoCatchBase64= function(operation, time, callbacks)
```


参数描述

表 6-95 参数说明

参数名	类型	可选/必选	描述
operation	int	必选	进行的操作类型，0为开始录屏(base64编码)，1为停止录屏(base64编码)
time	int	必选	操作类型为开始录屏时，传入定时录屏时间，最大不超过10秒，操作类型为结束录屏时传入0
callbacks	**Callback**	必选	回调方法。

表 6-96 Callback

参数名	类型	可选/必选	描述
response	function	必选	回调方法。

表 6-97 回调方法的入参

参数名	类型	描述
description	String	当前请求描述。
base64	String	视频的base64编码，录屏成功时返回
result	Number	设置结果。0表示成功，1表示失败，录屏失败时返回。
rsp	Number	内部消息编号。

说明

回调方法的入参示例：

```
{
  "description" : "OECatchVideo",
  "result" : 1,
  "rsp" : 67762
}

{
  "description" : "OECatchVideo",
  "base64" : "xxx",
  "rsp" : 67762
}
```

使用示例

```
function catchVideoBase64(operation){
    var time =  document.getElementById('CatchVideoTime').value;
    if (time != "" || operation == 3){
        this.global_openEye_SDK.openEyeCall.videoCatchBase64(operation, parseInt(time), { response:
startVideoCatchResponse })
    }
}

function startVideoCatchResponse(data){
    console.log(data);
    if (data.result == 0) {
        console.info("VideoCatch Success");
    } else {
        console.error("VideoCatch failed");
        console.error(data);
    }
}
```


7 错误码列表

模块	错误码（十进制值表示）	含义说明
openeyeSDK	0	接口调用成功
openeyeSDK	非0值	调用失败或者其他错误,需参考返回值中的其他描述

