Webrtc SDK 1.0 接口文档

# 需求来源

线上不断有客户反映js sdk不稳定，压缩过的文件1.2M依然过大，部分异常无法准确定位

# 存在问题

* Sip流程交互复杂，部分状态通过MESSAGE消息给出，包括登录、呼叫状态等，部分状态从sip信令中直接获取，不统一
* 外呼流程采取的回拨模式，向呼叫中心发送MESSAGE发起的双向回拨
* Sdk源码复杂，部分源码压缩过，不能准确定位问题，不好扩展

# 已有版本流程

## 登录流程



通过messge给出的登录状态，没从sip信令中获取

## 外呼流程

外呼采取的发message到呼叫中心，呼叫中心发起的双向回拨，通话状态通过message给出，没从sip信令中获取

## 呼入流程



呼入流程正常

# 详细设计

* 新版本采用sip.js 库替换原来的 telelive.js 库，新库相对简单
* 登录流程状态直接从sip信令中获取，不再重新发送message
* 外呼流程简化



* 私有消息全部从message的body给出，包括需要透传的用户数据，不再修改标准的sip头域

**初始化接口**

1. 函数名称：UCSConnect.init()
2. 函数描述：初始化JS模块，初始化回调函数
3. 返回值：void
4. 参数：

|  |  |
| --- | --- |
| callback | 填入相应的回调函数，可参考下方示例 |

1. 示例

当实现来电回调接口后，当有电话呼入时就会执行回调函数

var callback = {

//来电回调接口

OnIncomingCall: function(callid, caller, message) {

},

//呼叫被释放回调接口

OnCallGiveUp: function(message, cause) {

},

};

UCSConnect.init(callback);

**登录接口**

1. 函数名称：UCSConnect.login()
2. 函数描述：登录到云之讯平台,返回值代表消息发送成功或者失败。登陆成功和失败见相应回调函数
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| userName | 用户名 |
| password | 密码 |
| displayName | 用户昵称 |

5．示例

下面参数分别为用户输入的用户名和密码，在初始化函数中配置登陆成功和失败的回调函数来处理登陆的结果。（发起呼叫的逻辑需在登陆成功后，即：登陆接口回调接口成功）

<input type=*"button"* id=*"login"* value=*"登陆"*

onclick='UCSConnect.login(username.value,password.value,displayName.value)'>

**登录结果回调接口**

1. 函数名称：OnLoginRet()
2. 函数描述：用户登陆到平台的结果。
3. 返回值：void
4. 参数：

|  |  |
| --- | --- |
| message | SIP信令 |
| ret | 0: 成功  1: 登录超时  2：密码错误  3: 账号或密码为空  99：其他 |

1. 示例

回调接口在初始化函数中实现

// 登陆成功回调函数 响应码 描述 0: 成功（SUCCESS） 1: 超时（TIMEOUT）2：密码非法（INVALID\_PASSWORD）3: 账号或密码为空（USER\_NULL）99：其他（OTHER）

OnLoginRet: function(message，ret){

}

**登出接口**

1. 函数名称：UCSConnect.logout()
2. 函数描述：登出云之讯平台
3. 返回值：void
4. 参数 无
5. 示例 同登录接口

**登出结果回调接口**

1. 函数名称：OnLogoutRet()
2. 函数描述：用户登出平台的结果。
3. 返回值：void
4. 参数：

|  |  |
| --- | --- |
| message | SIP信令 |
| ret | 0: 成功  1: 失败 |

1. 示例：

//退出结果回调 0成功 1 失败

OnLogoutRet: function(message，ret){

}

**发起呼叫**

1. 函数名称：UCSCall.SendOutCall()
2. 函数描述：发送消息到PAAS平台，平台呼叫坐席，然后呼叫用户，采用SIP MESSAGE消息发送。
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| called | 被叫号码，支持 01088888888。(呼叫被叫时，需要在号码前加前缀77，比如呼叫的号码是18807559988，则 called 为7718807559988，固话01081234567，则called为7701081234567) |

1. 示例：UCSCall.SendOutCall(Called);

**通话状态通知回调**

1. 函数名称：OnOutCallStatus(message, ret)
2. 函数描述：通话状态回调接口
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| message | SIP信令 |
| ret | 1：拨号中  2：被叫接听  3：被叫拒接  4：通话挂断  5：主叫取消  6：通话结束  7：异常操作  99：未知 |

5. 示例：

OnOutCallStatus: function(message, ret) {

var str\_ret;

if (ret === 1) {

str\_ret = '拨号中';

} else if (ret === 2) {

str\_ret = '被叫接听';

} else if (ret === 3) {

str\_ret = '被叫拒接';

} else if (ret === 4) {

str\_ret = '通话挂断';

} else if (ret === 5) {

str\_ret = '主叫取消';

} else if (ret === 6) {

str\_ret = '通话结束';

} else if (ret === 7) {

str\_ret = '异常操作';

}

}

**来电回调接口**

1. 函数名称：OnIncomingCall()
2. 函数描述：来电回调接口，收到此回调后会再收到一条通话状态回调通知
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| caller | 主叫号码 |
| callID | callID |
| message | SIP信令 |

1. 示例：

OnIncomingCall: function(callID, caller, message) {

printLog('收到来电\n' + message);

}

**接听来电接口**

1. 函数名称：UCSCall.AnswerCall()
2. 函数描述：接听来电
3. 返回值：void
4. 参数：无
5. 示例：UCSCall.AnswerCall();

**拒绝来电接口**

1. 函数名称：UCSCall.CallReject()
2. 函数描述：释放呼叫接口
3. 返回值：void
4. 参数：无
5. 示例：UCSCall.CallReject();

**释放呼叫接口**

1. 函数名称：UCSCall.CallGiveup()
2. 函数描述：释放呼叫接口
3. 返回值：void
4. 参数：无
5. 示例：UCSCall. CallGiveup ();

**呼叫被释放回调接口**

1. 函数名称：OnCallGiveup()
2. 函数描述：呼叫释放回调接口
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| message | SIP信令 |
| cause | 挂断原因 |

1. 示例：

OnCallGiveup: function(message, cause) {

var str\_ret = '通话已结束';

printLog(str\_ret + "\n" + message);

}

**异常事件回调接口**

1. 函数名称：OnExcepting()
2. 函数描述：异常事件回调
3. 返回值：void
4. 参数

|  |  |
| --- | --- |
| message | SIP信令 |
| cause | 异常原因 |

1. 示例：

onExcepting: function(message, cause) {

var str\_ret = '异常事件';

//printLog(str\_ret + "\n" + message + '\ncause:' + cause);

}