# 远程组件浏览器版使用说明及接口说明

[远程组件浏览器版使用说明及接口说明 1](#_Toc71902236)

[1. 系统架构 2](#_Toc71902237)

[2. 功能说明 2](#_Toc71902238)

[3. 浏览器远程控件安装包 3](#_Toc71902239)

[4. controlengine.js 4](#_Toc71902240)

[5. niuniuremote.js 4](#_Toc71902241)

[6. 远程组件初始化接口 4](#_Toc71902242)

[7. 设置扩展参数信息 5](#_Toc71902243)

[8. 设置直连相关信息 6](#_Toc71902244)

[9. 远程信息初始化接口 6](#_Toc71902245)

[10. 结束远程 7](#_Toc71902246)

[11. 断开连接 7](#_Toc71902247)

[12. 获取当前远程状态 7](#_Toc71902248)

[13. 获取控件版本 8](#_Toc71902249)

[14. 屏幕录像 8](#_Toc71902250)

[15. 尝试启动控件（不含业务） 9](#_Toc71902251)

[16. 事件回调函数绑定 9](#_Toc71902252)

[17. 在客服或IM系统中，要集成远程组件 10](#_Toc71902253)

本文档旨在介绍在浏览器上如何集成远程组件，并且与具体业务系统进行对接。

## 系统架构

## 功能说明

远程组件的使用场景较多，最常用的是使用在在线客服及即时通讯系统中，使用的场景大致如下，两用户（A与B）正在使用IM或客服系统进行沟通，A用户想要协助B用户处理一些事情，但必需要在B用户的电脑上完成；这时候A用户直接远程控制 B用户的电脑则会非常方便。

使用本组件，A用户与B用户的交互流程大致如下：

1) A用户与B用户对远程信息进行初始化，如远程服务器地址，由A与B构成的唯一远程ID等信息

2) A通过客服系统向B发出远程控制请求指令

3) B用户接受A用户的请求，发出响应指令；同时调用远程组件进行远程连接

4) A用户收到B用户的响应指令，调用远程组件进行远程连接

5) A与B建立远程连接，进行服务

6) 远程功能使用完成，结束远程，断开连接

远程组件浏览器版本主要包含如下几部分信息：

## 浏览器远程控件安装包

 包含远程组件浏览器控件的核心文件，实现程序安装与信息注册，以供浏览器调用。

## controlengine.js

 用于封装浏览器控件js端的核心逻辑，完成控件调用以及回调函数响应的基础实现（集成过程中，此文件原样引入即可，无需修改）。

## niuniuremote.js

 用于封装远程组件所对应的接口的js端调用入口，同时提供函数返回值，远程组件内部状态通知等事件的封装（集成过程中，此文件原样引入即可，无需修改）。

 在集成过程中，用户页面主要与此文件中的js函数及事件进行交互，以驱动整个远程流程正常进行。

以下对niuniuremote.js中的函数以及事件接口进行说明：

## 远程组件初始化接口

this.InitRemote = function(myid, peerid, myname, peername, callback)

此接口用于在要启动远程前，先将要远程的双方的信息告知远程组件，以便进行初始化（此接口在页面中一般只调用一次）。

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 描述  | 备注 |
| myid | 字符串 | 我方标识号 | 这两个信息非常重要，组件需要根据这两个ID来形成一个唯一的identifier，以便在建立连接时能够正确匹配（正常情况下，只需以当前用户在业务系统中的唯一ID表示即可） |
| peerid | 字符串 | 对方标识号 |
| myname | 字符串 | 我的昵称 | 非必需 |
| peername | 字符串 | 对方昵称 |  |
| callback | 函数 | 接收一个obj对象，呈现调用初始化函数的返回值  | 非必需 |
|  |  |  |  |

此函数调用时，如果控件尚未安装或不可用，则会触发ShowDownload函数，提示用户进行下载安装。

## 设置扩展参数信息

this.SetParam = function(paramFlag, paramValue, callback)

此接口用于在远程前，进行若干扩展信息的设置，未来有扩展参数，也将通过此接口来进行设置；注意，此浏览器接口所设置的参数值均为utf8字符串。

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 描述  | 备注 |
| paramFlag | 整型 | 类型值枚举 | 详情见下表 |
| paramValue | 字符串 | 参数值 | 请使用unicode编码 |
| callback | 函数 | 接收一个obj对象，呈现调用函数的返回值  | 非必需 |

每一个paramFlag对应的参数详情信息如下:

*var eSetSsl = 1; //是否数据通道走SSL*

*var eSetSslPort = 2; //设置SSL模式下的端口号*

*var eSetAcceptDirect = 3; //设置是否可接受直连*

*var eSetEnableUpnp = 4; //设置是否启用upnp*

*var eSetUploadRecordUrl=5; //上报远程记录的接口地址*

*var eSetUploadMp4Url=6; //上报屏幕录像的地址*

*var eSetSessionId=7; //设置传话ID,唯一的标识一次客服与客户间的传话，目前用于上传*

*eSetForbidFileTransfer=8, //本地禁止文件传输*

*eSetLimitFileTransferSpeed=9, //本地限制文件传输速度（从被下载端限制）*

*eSetChangeRemoteType=11, //在远程过程中设置切换是观看还是控制*

|  |  |  |
| --- | --- | --- |
| 类型值 | 类型参数描述 | 备注 |
| 1 | 是置数据通道是否走SSL | 默认为0，如果设置后，数据通道将通过SSL与中转服务器通讯（此时不需要传递identifier） |
| 2 | 设置SSL模式下的端口号 | 当eSetSsl设置为1时有效 |
| 3 | 设置是否可接受直连 | 默认为0，不支持直连 |
| 4 | 设置是否启用内网穿透 | 默认为0，不支持穿透 |
| 5 | 设置远程状态上报的接口地址 | 支持http及https如果不设置地址或sessionid，则不会上传 |
| 6 | 设置远程录像的上报的接口地址 | 支持http及https如果不设置地址或sessionid，则不会上传 |
| 7 | 设置上传的标识ID | 应该与唯一的会话保持一致 |
| 8 | 设置是否禁用文件传输 | 在服务端开启文件传输的前提下生效，用于控制用户级是否禁用文件传输 |
| 9 | 设置文件传输的最大速度 | 两端均要设置，实际是在被下载端控制 |
| 11 | 在远程过程中设置切换是观看还是控制 | 1：控制或被控制2：观看或被观看此参数只有在远程已经建立的情况下生效，用于在远程过程中修改观看与控制的权限 |

具体的接口地址的参数说明，参见《利洽科技远程组件记录与视频上传接口说明》。

## 设置直连相关信息

this.InitDirectInfo = function(isSSL, pempwd, callback)

此接口用于设置端到端直连的相关配置信息（如果不调用将不会尝试直连）。

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 描述  | 备注 |
| isSSL | 整型 | 直连是否使用SSL连接(请保持与SetParam的SSL标志一致) | 1与0 |
| certpwd | 字符串 | 证书的密码 | utf8编码 |
| callback | 函数 | 接收一个obj对象，呈现调用函数的返回值  | 非必需 |

## 远程信息初始化接口

this.StartControl = function (asViewer, isControl, quality, callback)

此接口用于在用户双方都确定要建立远程时，双方在较为接近的时间范围内分别调用自己端的启动接口（双方一定要在业务系统中协商好后方可调用此接口，如果双方连接的时间差超过30秒，将无法建立连接）。

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 描述  | 备注 |
| asViewer | 整型 | 用于指定是作为控制端还是被控制端 | 1：表示作为控制端0：表示作为被控制端此处一定要设置正确，不同两端同时指定成同样的值，否则将无法建立远程。 |
| isControl | 整型 | 作为控制端时，是否处于控制模式 | 1：表示控制模式0：表示观看模式 |
| quality | 整型 | 作为控制端时，远程图像的默认质量 | 默认为1（尚需完善） |
| callback | 回调函数 | 远程服务器端口 | 非必需 |
|  |  |  |  |
|  |  |  |  |

此函数调用时，如果控件尚未安装或不可用，则会触发ShowDownload函数，提示用户进行下载安装。

## 结束远程

this.StopRemote = function(callback)

此接口用于结束当前远程（由myid以及peerid组成的唯一ID所标识），但不断开连接。

## 断开连接

this.DisConnect = function(callback)

此接口用于结束当前远程，同时断开连接（由myid以及peerid组成的唯一ID所标识）。

## 获取当前远程状态

this.GetStatus = function(callback)

此接口用于获取当前远程的状态（由myid以及peerid组成的唯一ID所标识）。

在callback的返回信息中，返回一个整型数 retValue，具体表示信息如下：

retValue /100000: 表示当前是否正在重连

(retValue / 100000)%10： 表示当前的网络连接状态：

var RemoteConn\_None = 0; //未连接

var RemoteConn\_Connecting = 1; //正在连接中

var RemoteConn\_Connected = 2; //已连接

retValue % 100000 % 10 ：表示当前的远程控制状态：

var RemoteNone = 0; //未建立远程

var VNCViewer = 1; //正在控制中或正在建立远程中

var VNCServer = 2; //正在被控制中或正在建立远程中

retValue % 100000 / 10 ：表示当前的远程过程中的观看与控制状态：

eControl\_Control=1, //控制或被控制中

eControl\_Watch=2, //观看或被观看中

**按位表述如下：**

//ab000xy 7位大小的整数

//百万位为1，表示在重连中

//十万位表示连接状态，0，未连接，1，连接中，2，已经连接

//万、千、百三位目前未使用

//十位表示当前的远程类型，是不是在控制中（个位为0时，此位一定为0）； 0表示未知；1表示控制或被控制中；2表示观看或被观看中

//个位，表示远程类型 0未远程；1控制中；2被控制中

## 获取控件版本

this.GetVersion = function(callback)

此接口用于在控件刚加载成功时，获取控件版本号；便于在控件需要升级时做版本号比对。

## 屏幕录像

此接口用于启动或停止屏幕录像。

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 描述  | 备注 |
| eventType | Int | 用于标识当前是开始录像还是停止录像 | 1: 开始录像2: 录像恢复3: 录像暂停4: 结束录像 |
| fps | int | 录像的FPS | 此值建议为3当eventType为1时有效 |
| quality | Int | 录像的质量 | 10至46之间，值越大质量越差；建议取值26-36当eventType为1时有效 |
| Showtime | int | 录像是否显示当前时间 | 0: 不显示1: 显示当eventType为1时有效 |
| recordType | int | 录像的行为控制 | 1: 录取远程窗口所在屏幕2: 只录取远程窗口本身区域3: 录取当前鼠标所在屏幕4: 录取整个屏幕，有多个扩展屏幕就录制多个此值建议取1当eventType为1时有效 |

## 尝试启动控件（不含业务）

this.StartEngine = function()

此接口用于仅启动浏览器控件引擎，而不启动具体的远程业务。

主要是在页面刚加载完成时，作为提示用户启动或下载后点击的事件响应。

## 事件回调函数绑定

self.PluginLoaded //控件加载成功时调用的函数

this.PluginUnload //浏览器控件断开时调用的函数

this.OnControlNotice //远程组件内部消息通知函数 （当连接建立、断开等事件发生时，会触发此函数）

this.doShowDownload //当需要提示下载组件时将被调用，用于在UI上呈现

## 在客服或IM系统中，要集成远程组件

参考随文档一并发布的示例即可：

1) 关注本地JS与远程组件控件的交互流程

2) 业务系统中双方在远程前以及远程过程中的协商需要另外在业务系统中实现（示例中采用了socket.io来进行展示，但不包含服务端代码，集成时需要隔离掉）

3) 参考此示例时，请重点关注imremote.js文件，内部已经包含了大部分业务的封装