

# AnyChat SDK for Web

## 开发流程指南

(版本: V9.0)



广州佰锐网络科技有限公司

GuangZhouBaiRui Network Technology Co.,Ltd.

<http://www.bairuitech.com>

<http://www.anychat.cn>

2021 年 6 月

# 目录

一、	简介.....	3
1.1	面向的读者.....	4
1.2	获取 ANYCHAT QUEUE FOR WEB.....	4
1.3	技术支持.....	5
二、	编写说明.....	6
三、	工程准备.....	7
3.1	安装 ANYCHAT WEB 插件.....	7
3.2	导入 SDK 文件.....	8
四、	基本流程.....	9
4.1	监听基本事件.....	9
4.2	初始化 SDK.....	10
4.3	连接、登录服务器.....	11
五、	音视频交互.....	13
5.1	打开本地音视频.....	13
5.2	关闭本地音视频.....	13
5.3	请求远程音视频.....	14
5.4	关闭远程音视频.....	14
六、	业务排队.....	15
6.1	初始化业务对象身份.....	15
6.2	显示/进入/离开营业厅.....	17
6.3	显示/进入/离开队列.....	19
6.4	坐席服务.....	21
七、	视频呼叫.....	23
7.1	视频呼叫请求.....	23
7.2	视频呼叫回复.....	24
7.3	视频呼叫开始.....	28
7.4	视频呼叫结束.....	30
八、	资源释放.....	32
九、	附录.....	33
9.1	HELLOANYCHAT 界面.....	33
9.2	ANYCHATQUEUE 界面.....	33

# 一、简介

AnyChat SDK(AnyChat 音视频互动开发平台)是一套跨平台的音视频即时通讯解决方案，基于先进的 H.264 视频编码标准、AAC 音频编码标准与 P2P 技术，支持高清视频，整合了佰锐科技在音视频编码、多媒体通讯领域领先的开发技术和丰富的产品经验而设计的高质量、宽适应性、分布式、模块化的网络音视频互动平台。

基于 Web 的客户端 SDK 应用于基于 Web 的应用环境，提供了一套基于 JavaScript 的应用程序接口，支持目前市场上主流的浏览器（参考：[浏览器兼容性测试记录](#)），您可以通过该套 SDK API 接口实现快速开发基于音视频通讯交互功能的 Web 应用程序，主要提供的功能如下：

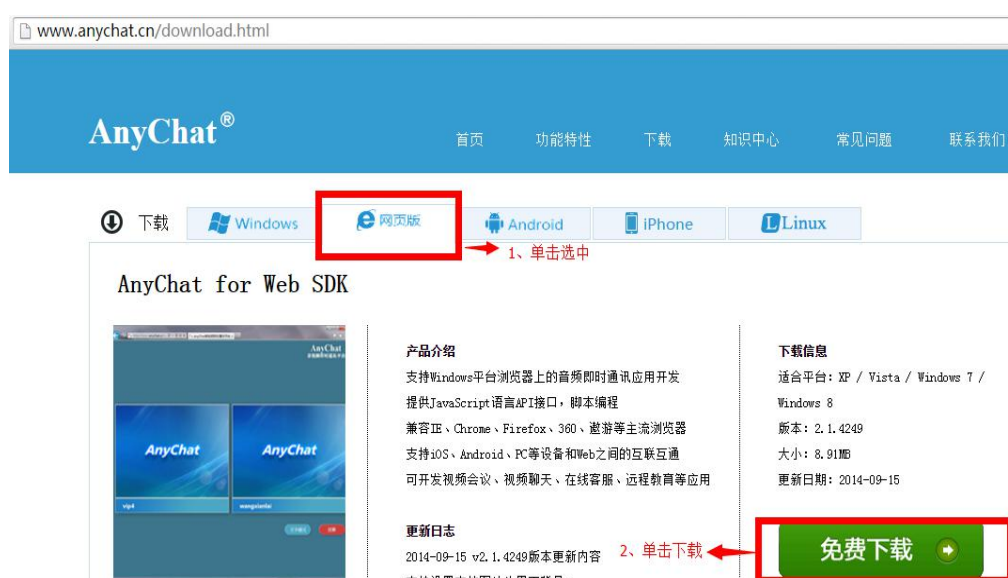
- 音视频即时通讯：提供语音、视频一对一、一对多的实时通讯，支持高清视频和高品质音频效果。
- 录像：支持针对单个人的音视频录制、整个视频通话过程内容的合成音视频录制以及集中服务器保存录制
- 抓拍：可对本地视频和正在视频的对象进行抓拍；
- 文字聊天：支持多用户之间的文字交流；
- 透明通道：提供客户端之间、客户端跟服务器之间的数据通讯能力；
- 文件传输：支持客户端之间、客户端跟服务器之间的文件传输功能，支持断点续传；
- 动态设置音视频参数：提供音视频参数设置的接口，可以根据需要动态设置分辨率、码率、帧率等视频参数，满足各种应用场景的需求；
- 外部音视频输入：支持非标准采集设备以外的音视频源输入，满足更多的应用场景；
- 集成第三方外部音视频编解码器：可集成第三方音视频编解码器，满足特殊环境下面的硬件编解码要求；
- 业务排队：提供自定义营业区域、队列功能，实现客户排队、坐席为队列中客户提供服务的功能；

## 1.1 面向的读者

《AnyChat Queue for Web 开发流程指南》文档是提供给具有一定的 JavaScript 编程经验和了解网页开发的读者使用，不要求具备音视频开发方面的经验。您在使用遇到任何问题，都可以通过访问 [bbs.anychat.cn](http://bbs.anychat.cn) 反馈给我们。

## 1.2 获取 AnyChat Queue for Web

您可在 AnyChat 的产品官方网站下载到最新版的 AnyChat for Web SDK，下载地址为: <http://www.anychat.cn/download.html>，如下图所示：



AnyChat for Web SDK 包提供了 Web SDK 插件、开发指南、Demo 源代码、JavaScript 接口，其解压之后的目录结构如下所示：

----bin	AnyChat for Web SDK 插件（安装包）
----doc	客户端开发指南
----src	Demo 程序源代码
----sdk	JavaScript 库
----server	服务器程序

## 1.3 技术支持

在您使用本 SDK 的过程中，遇到任何困难，请与我们联系，我们将热忱为您提供帮助。

您可以通过如下方式与我们联系：

- 1、在线论坛：<http://bbs.anychat.cn/>
- 2、知识中心：<http://www.anychat.cn/faq/>
- 3、官方网站：<http://www.anychat.cn>
- 4、电子邮件：[service@bairuitech.com](mailto:service@bairuitech.com)
- 5、24 小时客服电话：+86 （020） 85276986、38109065、38103410

## 二、编写说明

本指南的编写是为了帮助使用 AnyChat for Web SDK 的用户快速地搭建 SDK 开发环境、熟悉 SDK 开发流程、掌握 SDK 开发功能接口而编写的。

其中“工程准备”、“基本流程”、“音视频交互”三章的内容是基于 src\HelloAnyChat 目录中提供的 AnyChatWebDemo 工程来编写说明的，涉及到的开发环境配置、以及相关代码说明可以参考 AnyChatWebDemo 工程源码。

其中“业务排队”章节内容是基于 src 目录中提供的 AnyChatQueue 工程来编写说明的，涉及到的开发环境配置、以及相关代码说明可以参考 AnyChatQueue 工程源码。

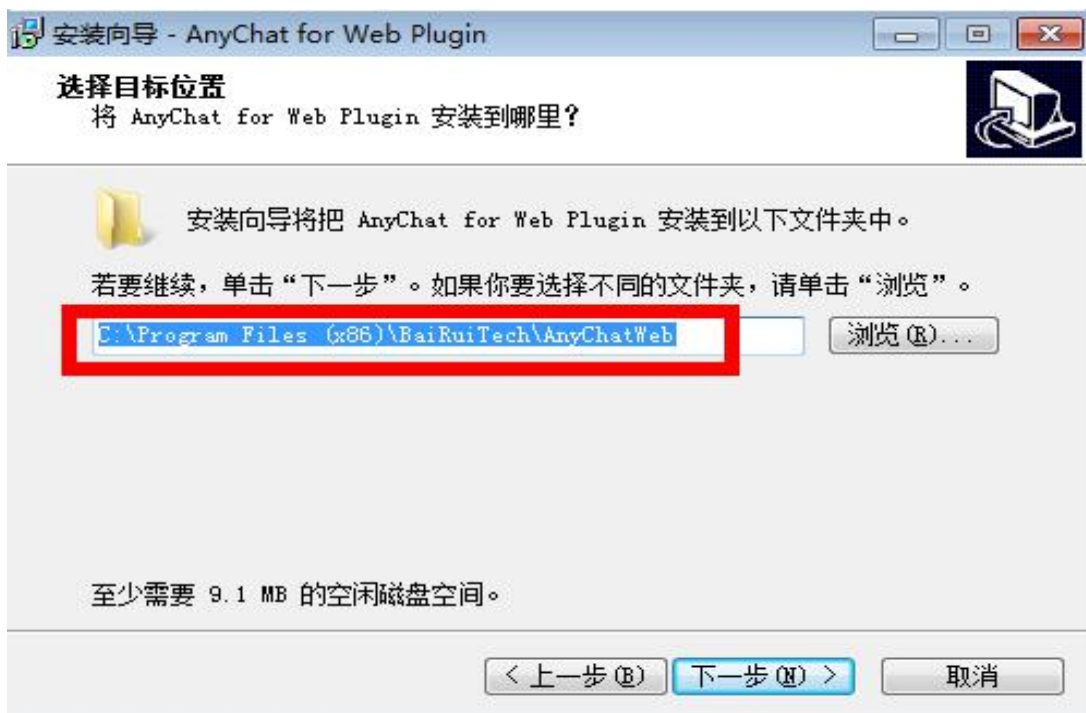
## 三、工程准备

JavaScript 的开发工具有很多，开发者可根据自己的喜好进行选择。在此，我们推荐开发者使用 dreamwaver 作为自己的开发工具，本套开发指南也是针对 dreamwaver 开发环境下进行编写的。

### 3.1 安装 AnyChat Web 插件

AnyChat for Web SDK 包中提供了一个 ActiveX 插件（如果您自己有 ActiveX 代码签名证书，可对 SDK 包 bin\AnyChatWeb 目录中的“npnanychatweb.dll”和“npvideoshowctrl”插件进行签名，具体操作参考 bin 下面的 readme.txt 文件），AnyChat Web SDK 提供的所有 JavaScript 接口都是基于这个插件实现。在使用 AnyChat Web SDK 接口进行通讯之前，需要确保插件已经安装成功。安装插件的步骤如下：

- (1) 打开解压后的 SDK 包下的 bin 目录，双击 AnyChatWebSetup.exe 程序进行安装，安装的过程中可以选择安装路径，如下图所示：



- (2) 按照提示，单击“下一步”按钮，即可完成插件的安装。

## 3.2 导入 SDK 文件

- (1) 将解压后的 sdk 包下 src\sdk 目录的 anychatsdk.js 文件拷贝到网页工程的相应的目录下面，我的目录结构如下图所示：



- (2) 在需要使用 AnyChat Web SDK 接口的网页中引入 anychatsdk.js,参考代码如下：

```
<head>
<title>AnyChat for Web SDK Demo</title>
<!-- 加载 AnyChat for Web SDK 库-->
<script language="javascript" type="text/javascript"
src="./javascript/anychatsdk.js" charset="GB2312"></script>
</head>
```



## 四、基本流程

在工程准备好了之后，需要实现以下基本流程，才能调用音视频交互等其他功能接口，才能调用音视频交互等其他功能接口。

### 4.1 监听基本事件

监听“连接服务器、用户登录、进入营业厅、进入队列、视频呼叫、视频通话”等事件。

(1) 将 SDK 包下 `src\AnyChatQueueDemo\javascript\anychatevent.js`、`anychatobject.js` 拷贝到网页工程相应目录下，并在网页中加载 `anychatevent.js`、`anychatobject.js` 文件，参考代码如下：

```
<head>
<script      language="javascript"      type="text/javascript"
src="./javascript/anychatevent.js" charset="GB2312"></script>
<script      language="javascript"      type="text/javascript"
src="./javascript/anychatobject.js" charset="GB2312"></script>
</head>
```

(2) 在 `anychatevent.js` 的函数体中可以添加上层需要处理的代码，不需要的函数可以删除。在相应的条件下内核会自动执行 `anychatevent.js` 的相应函数，参考代码如下：

```
//业务对象事件通知
function OnAnyChatObjectEvent(dwObjectType, dwObjectId, dwEventType, dwParam1,
dwParam2, dwParam3, dwParam4, strParam) {
    console.info("OnAnyChatObjectEvent(dwObjectType=" + dwObjectType + ",
dwObjectId=" + dwObjectId + ", dwEventType=" + dwEventType + ")");
    switch(dwEventType) {
        case ANYCHAT_OBJECT_EVENT_UPDATE:
            OnAnyChatObjectUpdate(dwObjectType, dwObjectId);           break;
        case ANYCHAT_AREA_EVENT_ENTERRESULT:
            OnAnyChatEnterAreaResult(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_AREA_EVENT_LEAVERESULT:
            OnAnyChatLeaveAreaResult(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_AREA_EVENT_USERENTER:
            OnAnyChatUserEnterArea(dwObjectType, dwObjectId, dwParam1, dwParam2, dwParam3,
dwParam4); break;
        case ANYCHAT_AREA_EVENT_USERLEAVE:
            OnAnyChatUserLeaveArea(dwObjectType, dwObjectId, dwParam1, dwParam2);
            break;
        case ANYCHAT_QUEUE_EVENT_STATUSCHANGE:
            OnAnyChatQueueStatusChanged(dwObjectType, dwObjectId);       break;
        case ANYCHAT_QUEUE_EVENT_ENTERRESULT:
            OnAnyChatEnterQueueResult(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_QUEUE_EVENT_LEAVERESULT:
            OnAnyChatLeaveQueueResult(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_QUEUE_EVENT_USERENTER:
            OnAnyChatUserEnterQueue(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_QUEUE_EVENT_USERLEAVE:
            OnAnyChatUserLeaveQueue(dwObjectType, dwObjectId, dwParam1); break;
        case ANYCHAT_AGENT_EVENT_STATUSCHANGE:
            OnAnyChatAgentStatusChanged(dwObjectType, dwObjectId, dwParam1);
            break;
        case ANYCHAT_AGENT_EVENT_SERVICENOTIFY:
            OnAnyChatServiceStart(dwParam1, dwParam2);                   break;
        case ANYCHAT_AGENT_EVENT_WAITINGUSER:    OnAnyChatAgentWaitingUser();
            break;
        default:
            break;
    }
}
```

## 4.2 初始化 SDK

加载插件资源，其他的功能接口都必须在初始化成功之后才能正常使用。可以放在需要使用 AnyChat 功能的网页 onload 事件中延时(setTimeout)调用。如果初始化失败，可能是没有安装插件或者插件版本太老，可以提示客户需要安装插件并提供插件下载链接，并启动定时器不断调用初始化，直到用户安装插件成功。参考代码如下：

```
function loadAnyChat(){
    setTimeout(function () {
        //刷新网页加载插件列表
        if (navigator.plugins && navigator.plugins.length) {
            window.navigator.plugins.refresh(false);
        }
        var errorcode=BRAC_InitSDK(0);
        if(errorcode==0){
            //初始化成功，清除插件安装检测定时器
            if(mRefreshPluginTimer != -1){
                clearInterval(mRefreshPluginTimer);
            }
        }else{
            if(errorcode==GV_ERR_PLUGINNOINSTALL){
                //初始化失败，需要安装插件,提供插件下载链接给客户
            }
            else if(errorcode==GV_ERR_PLUGINOLDVERSION){
                //初始化失败，当前插件的版本过低，请下载安装最新版本
            }
            if(mRefreshPluginTimer == -1) {
                mRefreshPluginTimer = setInterval(function(){ LogicInit();
                }, 1000);
            }
        }
    }, 500);
}
window.onload = loadAnyChat;
```

## 4.3 连接、登录服务器

使用 AnyChat 功能的通讯功能，必须要先连接登录 AnyChat 的通讯服务器，这里以我们对外公开测试服务器地址 `demo.anychat.cn` 为例，参考代码如下：

```
// 连接服务器,第一个参数为你需要连接的视频服务器地址，如果您部署视频服务器的地址为  
192.168.1.8，则传入这个地址  
BRAC_Connect("demo.anychat.cn", 8906);  
//登录服务器  
BRAC_Login("AnyChat", "", 0);
```

这两个流程的执行都是异步的操作，会触发相应的 `OnAnyChatConnect`、`OnAnyChatLoginSystem` 函数。

## 五、音视频交互

AnyChat for Web SDK 为开发者提供了便捷的建立音视频通讯的接口，通过以下几步操作，即可在您的应用中集成音视频交互功能。需要注意的是只有在同一个房间内的用户才能进行音视频通讯。

### 5.1 打开本地音视频

打开本地/远程音视频数据需要在进入房间成功之后才有效，这里是列举了打开远程视频的方法，参考代码如下：

```
// 控制视频打开关闭
function startVideo(uid, videoID, videoType, state) {
    /**视频操作*/
    BRAC_UserCameraControl(uid, state);
    /**语音操作*/
    BRAC_UserSpeakControl(uid, state);
    /**设置视频显示位置*/
    BRAC_SetVideoPos(uid, videoID, videoType);
}
// 调用方式设置远程视频显示位置并打开(调用方式)
startVideo(mSelfUserId, GetID("localVideoPos"), "ANYCHAT_VIDEO_LOCAL", 1);
```

### 5.2 关闭本地音视频

打开本地音视频后，可以在音视频交互的过程中选择关闭本地音视频。同时，还可以在关闭之后重新打开本地音视频（参考 5.1）；在音视频交互结束之后需要调用该操作，释放本地摄像头和音频采集设备，参考代码如下：

```
// 调用方式设置远程视频显示位置并关闭(调用方式)
startVideo(mSelfUserId, GetID("localVideoPos"), "ANYCHAT_VIDEO_REMOTE", 0);
```

## 5.3 请求远程音视频

在通话目标对象已经进入当前房间之后（参考 4.4），该操作才有效。调用 5.1 里的 `startVideo` 方法进行视频调用，参考代码如下：

```
// 调用方式设置远程视频显示位置并打开 (调用方式)
startVideo(mTargetUserId, GetID("remoteVideoPos"), "ANYCHAT_VIDEO_REMOTE",
1);
```

## 5.4 关闭远程音视频

请求远程音视频后，可以在音视频交互的过程中选择关闭远程音视频。同时，还可以在关闭之后重新请求远程音视频（参考 5.5）；在音视频交互结束之后需要调用该操作，释放远程音视频资源，参考代码如下：

```
// 调用方式设置远程视频显示位置并关闭 (调用方式)
startVideo(mTargetUserId, GetID("remoteVideoPos"), "ANYCHAT_VIDEO_REMOTE",
0);
```

## 六、业务排队

AnyChat for Web SDK 为开发者提供了实现业务队列及用户排队、座席为队列内的用户进行服务的接口，通过以下几步操作，即可在您的应用中集成业务排队功能。座席从队列中取出用户后，通过调用视频呼叫的接口实现音视频交互。

AnyChat 排队业务解决方案的介绍可以点击 “[AnyChat 提供业务排队整体解决方案](#)” 链接查看。

### 6.1 初始化业务对象身份

在 anychatevent.js 的 OnAnyChatLoginSystem 函数中设置登入者的身份，登录服务器后触发登录系统的函数后直接调用下面的方法，第二个参数指的是身份标识，dwAgentFlags=ANYCHAT\_OBJECT\_FLAGS\_AGENT;则表示为坐席，传入 0 则表示为客户，参考代码如下：

```
//初始化本地对象信息
function InitClientObjectInfo(mSelfUserId,dwAgentFlags) {

    AddLog("Initialize Client Object Information", LOG_TYPE_NORMAL);

    //初始化服务区域数组

    areaArrayIdx = 0;

    arealdArray = new Array();

    /**业务对象身份初始化*/
    BRAC_SetSDKOption(BRAC_SO_OBJECT_INITFLAGS, dwAgentFlags);
    var dwPriority = 10;
    /**客户端用户对象优先级*/
    BRAC_ObjectSetValue(ANYCHAT_OBJECT_TYPE_CLIENTUSER, mSelfUserId,
    ANYCHAT_OBJECT_INFO_PRIORITY, dwPriority);
    var dwAttribute = -1;
    BRAC_ObjectSetValue(ANYCHAT_OBJECT_TYPE_CLIENTUSER, mSelfUserId,
    ANYCHAT_OBJECT_INFO_ATTRIBUTE, dwAttribute);
    /**向服务器发送数据同步请求指令*/
    BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AREA, ANYCHAT_INVALID_OBJECT_ID,
    ANYCHAT_OBJECT_CTRL_SYNCDATA, mSelfUserId, 0, 0, 0, "");
}

// 客户端登录系统, dwUserId 表示自己的用户 ID 号, errorcode 表示登录结果: 0 成功, 否则为出
错代码, 参考出错代码定义
function OnAnyChatLoginSystem(dwUserId, errorcode) {
    if (errorcode == 0) {
        if (userType == 2) { //客服
            currentAgentID = dwUserId;
            dwAgentFlags=ANYCHAT_OBJECT_FLAGS_AGENT; //坐席标识
        } else if (userType==1) { //客户
            dwAgentFlags=0;
        }
        mSelfUserId = dwUserId;
        //身份信息设置
        InitClientObjectInfo(mSelfUserId,dwAgentFlags);
    }
}
```



## 6.2 显示/进入/离开营业厅

在业务对象身份初始化之后，需要调用 BRAC\_ObjectControl 接口发送服务区域数据同步请求指令同步已存在的服务区域(营业厅)数据，接口第二个参数为 -1 表示同步所有的营业厅。参考代码如下：

```
/**向服务器发送数据同步请求指令*/  
BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AREA, ANYCHAT_INVALID_OBJECT_ID,  
ANYCHAT_OBJECT_CTRL_SYNCDATA, mSelfUserId, 0, 0, 0, "");
```

每同步一个服务区域(营业厅)时，都会触发 anychatevent.js 的 OnAnyChatObjectUpdate 这个事件，在此事件中我们记录系统内存在的服务区域(营业厅)数量，保存在数组中，代码如下：

```
/**向服务器发送数据同步请求指令*/  
  
if(dwObjectType == ANYCHAT_OBJECT_TYPE_AREA) {  
  
    arealIdArray[areaArrayIdx] = dwObjectId;
```

在同步完成后会触发 anychatevent.js 的 ANYCHAT\_OBJECT\_EVENT\_SYNC DATAFINISH 这个事件，响应此事件处理显示营业厅的业务逻辑，参考代码如下：

```

//业务对象同步完成事件

function OnAnyChatObjectSyncDataFinish(dwObjectType, dwObjectId) {

    AddLog('OnAnyChatObjectSyncDataFinish(' + dwObjectType + ',' + dwObjectId + ')',
    LOG_TYPE_EVENT);

    if (dwObjectType == ANYCHAT_OBJECT_TYPE_AREA) {

        showServiceArea();

    }

}

//显示服务区域(营业厅)
function showServiceArea() {
    var areaName = "";
    var description = "";

    $("#loginDiv").hide(); //隐藏登录界面
    $("#enterRoom").show(); //显示大厅
    $("#enterRoom h2:eq(1)").text("营业厅列表");

    for (var idx in areaIdArray)
    {
        areaName = BRAC_ObjectGetStringValue(ANYCHAT_OBJECT_TYPE_AREA, areaIdArray[idx],
        ANYCHAT_OBJECT_INFO_NAME);
        description = BRAC_ObjectGetStringValue(ANYCHAT_OBJECT_TYPE_AREA, areaIdArray[idx],
        ANYCHAT_OBJECT_INFO_DESCRIPTION);

        if (queueListName == -1) {
            $("#poptip li").each(function (index) {
                if (areaIdArray[idx] == $(this).attr('dwObjectId')) { $(this).remove(); }
            });
            var createObj = $('<li dwObjectId="' + areaIdArray[idx] + '">' + '<p>' + areaName + '</p>' + '<p'
            class="description">' + description + '</p>' + '<p>' + '' + '</p>' + '<p>编号: ' +
            areaIdArray[idx] + '</p>' + '<p>' + '<a class="btn">进入</a>' + '</p>' + '</li>');
            createObj.css("background-color", colorArray[colorIdx]);
        }
    }
}

```

将营业厅显示在页面之后，可以调用 BRAC\_ObjectControl 接口，执行 ANYCHAT\_AREA\_CTRL\_USERENTER 方法进入某个营业厅。参考代码如下：

```
/**进入营业厅*/  
var errorcode = BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AREA, hallbusinessNum,  
ANYCHAT_AREA_CTRL_USERENTER, 0, 0, 0, 0, "");  
AddLog("BRAC_ObjectControl(" + ANYCHAT_OBJECT_TYPE_AREA + "," + hallbusinessNum  
+ "," + ANYCHAT_AREA_CTRL_USERENTER + ",0,0,0,0," + ")=" + errorcode, LOG_TYPE_API);
```

在执行成功后，会触发 ANYCHAT\_AREA\_EVENT\_ENTERRESULT 事件（用户进入营业厅事件）。

如果要退出某个营业厅，也可以调用 BRAC\_ObjectControl 接口，执行 ANYCHAT\_AREA\_CTRL\_USER LEAVE 方法进入某个营业厅。参考代码如下：

```
//离开营业厅  
var errorcode = BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AREA,  
hallbusinessNum, ANYCHAT_AREA_CTRL_USERLEAVE, 0, 0, 0, 0, "");  
AddLog("BRAC_ObjectControl(" + ANYCHAT_OBJECT_TYPE_AREA + "," +  
hallbusinessNum + "," + ANYCHAT_AREA_CTRL_USERLEAVE + ",0,0,0,0," + ")=" + errorcode,  
LOG_TYPE_API);
```

## 6.3 显示/进入/离开队列

身份为“客户”的用户在进入营业厅后，会触发函数 OnAnyChatEnterArea Result，在 OnAnyChatEnterAreaResult 事件（用户进入营业厅事件）中可以实现显示营业厅中的队列功能，会调用到 BRAC\_ObjectGetIdList、BRAC\_ObjectGetStringValue 等接口获取队列的信息，如队列 ID 列表、队列名称、队列描述信息、队列中当前人数等。参考代码如下：

```
//获取队列

var queueList =BRAC_ObjectGetIdList(ANYCHAT_OBJECT_TYPE_QUEUE);

for (var i in queueList) {

    var queueListId = parseInt(queueList[i]);

    //获取队列名称

    var queueName =

BRAC_ObjectGetStringValue(ANYCHAT_OBJECT_TYPE_QUEUE, queueListId,

ANYCHAT_OBJECT_INFO_NAME);

    //获取队列排队人数

    var queueLength =

BRAC_ObjectGetIntValue(ANYCHAT_OBJECT_TYPE_QUEUE, queueListId,

ANYCHAT_QUEUE_INFO_LENGTH);

    queueLength = queueLength < 0 ? 0 : queueLength;

    //获取队列信息

    var queueInfo =

BRAC_ObjectGetStringValue(ANYCHAT_OBJECT_TYPE_QUEUE, queueListId,

ANYCHAT_OBJECT_INFO_DESCRIPTION);

    $("#LOADING_GREY_DIV").hide(); //隐藏蒙层

    $('#poptip li[dwobjectid]').hide(); //隐藏服务厅

    $("#enterRoom h2:eq(1)").text(queueListName);
```

在已显示的队列中，通过调用 BRAC\_ObjectControl 接口，执行 ANYCHAT\_QUEUE\_CTRL\_USERENTER 方法进入某个队列。参考代码如下：

```
/**进入队列*/  
var errorcode = BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_QUEUE, queueid,  
ANYCHAT_QUEUE_CTRL_USERENTER, 0, 0, 0, 0, "");  
    AddLog("BRAC_ObjectControl(" + ANYCHAT_OBJECT_TYPE_QUEUE + "," + queueid + ","  
+ ANYCHAT_QUEUE_CTRL_USERENTER + ",0,0,0,0," + ")=" + errorcode, LOG_TYPE_API);
```

在进入队列后就开始排队，等待坐席的服务，在排队期间可以通过 `BRAC_ObjectGetValue` 接口获取队列的属性信息，如：获取当前队列人数、获取排在自己前面的用户数、自己在队列中的等待时间等。参考代码如下：

```
/**获取队列名称*/  
  
var queueName = BRAC_ObjectGetStringValue(ANYCHAT_OBJECT_TYPE_QUEUE, queueListId,  
ANYCHAT_OBJECT_INFO_NAME);  
  
/**获取队列排队人数*/  
  
var queueLength = BRAC_ObjectGetIntValue(ANYCHAT_OBJECT_TYPE_QUEUE, queueListId,  
ANYCHAT_QUEUE_INFO_LENGTH);
```

在等待的过程中，通过调用 `BRAC_ObjectControl` 接口，执行 `ANYCHAT_QUEUE_CTRL_USERLEAVE` 方法实现客户退出队列功能。参考代码如下：

```
/**离开队列*/  
var errorcode = BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_QUEUE, queueid,  
ANYCHAT_QUEUE_CTRL_USERLEAVE, 0, 0, 0, 0, "");
```

## 6.4 坐席服务

身份为“坐席”的用户在 ANYCHAT\_AREA\_EVENT\_ENTERRESULT 事件（用户进入营业厅事件）中进行显示视频服务页面的逻辑处理进入视频服务页面，在页面中点击开始服务按钮则会触发下面的函数：

```
$('#startService').click(function () {  
    /**客服开始服务*/  
    BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AGENT,mSelfUserId,ANYCHAT_AGENT_CTRL_S  
    ERVICEREQUEST, 0, 0, 0, 0, "");  
});
```

收到开始服务事件后，又会触发 anychatevent.js 中的 OnAnyChatServiceStart 回调，在该回调中会有呼叫者 id 和被呼叫者 id 的传入，可以在该函数中编写呼叫的逻辑

```
// 坐席服务开始  
  
function OnAnyChatServiceStart(dwAgentId, clientId, dwQueueId) {  
  
    AddLog('OnAnyChatServiceStart(' + dwAgentId + ',' + clientId + ',' + dwQueueId + ')',  
    LOG_TYPE_EVENT);  
  
    if (userType == 2 && mSelfUserId == dwAgentId) {  
  
        $("#LOADING_GREY_DIV span").hide();  
  
    }  
}
```

后面关于如何呼叫客户，具体参考第七章

## 七、视频呼叫

AnyChat for Web SDK 为开发者提供了视频呼叫的功能，实现两个用户之间（如客户与坐席）的视频呼叫请求（Request）、视频呼叫回复（Reply）、视频呼叫开始（Start）以及视频呼叫结束（Finish）等过程，可以形象理解为打电话的流程：拨号、等待、通话、挂断。

AnyChat 视频呼叫详细技术实现可以点击 “[AnyChat 视频呼叫业务逻辑详解](#)” 链接查看。

### 7.1 视频呼叫请求

视频呼叫请求由请求方（如坐席）向被服务的一方（如客户）发起，通过调用 **BRAC\_ObjectControl** 接口（视频呼叫控制接口）去发送 ANYCHAT\_AGENT\_CTRL\_SERVICEREQUEST 事件实现发送视频呼叫请求，参考代码如下：

```
//开始服务按钮事件(坐席)
```

```
$('#startService').off().on('click', function () {
```

```
    if (!startServiceTag) {
```

```
        startServiceTag = true;
```

```
        /**客服开始服务*/
```

```
        BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AGENT,
```

被服务的一方（如客户）接收到 **BRAC\_VIDEOCALL\_EVENT\_REQUEST** 事件后要实现是否接收请求的业务逻辑处理，参考代码如下：

```
//收到视频呼叫请求
function onVideoCallControlRequest(dwUserId, dwErrorCode, dwFlags, dwParam, szUserStr)
{
    $("#callLayer h4").text("收到服务请求");
    /**获取客服姓名*/
    var UserName = BRAC_GetUserInfo(dwUserId, USERINFO_NAME);
    $("#callLayer #queueMsg1").hide();
    $("#callLayer #queueMsg2").show();
    mTargetUserId=dwUserId;
    $("#callLayer #queueMsg2 p").html("客服 <b style=\"color:red;\">"+UserName+"</b> 请求与您视频通
话，是否接受? ");
}
```

## 7.2 视频呼叫回复

视频呼叫回复由被服务的一方（如客户）发起，通过调用 BRAC\_VideoCall Control 接口（视频呼叫控制接口）去发送 BRAC\_VIDEOCALL\_EVENT\_REPLY 事件实现发送视频呼叫回复；另外被服务的一方（如客户）也可以选择拒绝请求方（如坐席）的视频呼叫请求，参考代码如下：



```
// 排队信息确认框

$("#callLayer button.confirmMsg").click(function () {

    $("#callLayer h4").text("服务等待中");

    clearInterval(waitTimeSet); //清除排队时间计时器

    $('#callLayer').hide(); //隐藏排队信息窗口

    $("#queueMsg2").hide(); //隐藏客服呼叫信息

    //判断信息执行对应操作

    switch ($(this).text()) {

        case "取消排队":

            $("#enterRoom h2").text(queueListName);

            $("#LOADING_GREY_DIV span").text("取消排队中，请稍候.....");//等待蒙层文本填充

            //$("#LOADING_GREY_DIV span").text("");//清空等待蒙层
```

```
case "接受":

    $("#enterRoom h2").text("客服端视频窗口");

    AcceptRequestBtnClick();

    break;

case "拒绝":

    $("#enterRoom h2").text(queueListName);

    $("#poptip").show();

    $("#poptip li[queueid]").show(); //显示队列列表

    RejectRequestBtnClick();

    $("#roomOut").off().click(function(){
```

传入 BRAC\_ERROR\_VIDEOCALL\_REJECT 参数实现拒绝呼叫请求，参考代码如下：

```
//拒绝会话

function RejectRequestBtnClick() {

    /**目标用户拒绝会话触发*/

    BRAC_VideoCallControl(BRAC_VIDEOCALL_EVENT_REPLY,mTargetUserId,GV_ERR_SESSION_REFUSE,0,0,"");

    /**离开队列*/
```

请求方（如坐席）在收到 BRAC\_VIDEOCALL\_EVENT\_REPLY 事件后要根据被服务的一方（如客户）的回复情况实现不同的业务逻辑处理，参考代码如下：

```
//视频呼叫请求回复

function onVideoCallControlReply(dwUserId, dwErrorCode, dwFlags, dwParam, szUserStr)
{
    switch(dwErrorCode)
    {
        case GV_ERR_SUCCESS://成功的情况

            onSendVideoCallRequestSucess(dwUserId);

            break;

        case GV_ERR_SESSION_QUIT:

            /**离开队列*/

            BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_QUEUE, queueid,
ANYCHAT_QUEUE_CTRL_USERLEAVE,0 ,0,0,0,"");

            $('#callLayer').hide();

            $("#enterRoom h2").text(queueListName);

            $('#poptip').show();

            clearInterval(waitTimeSet);

            ForSession("用户主动放弃会话", true);
```

```
case GV_ERR_SESSION_TIMEOUT:

    $("#Initiative_Call_Div").hide();

    CancelCall();

    ForSession("会话请求超时",true);

    break;

case GV_ERR_SESSION_DISCONNECT:
```

## 7.3 视频呼叫开始

被服务的一方（如客户）通过调用 BRAC\_VideoCallControl 接口（视频呼叫控制接口）去发送 BRAC\_VIDEOCALL\_EVENT\_REQUEST 事件同意了请求方（如坐席）的视频呼叫请求，则系统会触发 BRAC\_VIDEOCALL\_EVENT\_START 事件，系统会自动的分配一个房间号，双方都需要进入该房间号，程序需要响应此事件实现开始视频呼叫的业务逻辑功能。参考代码如下：

```
//通话开始

function onVideoCallControlStart(dwUserId, dwErrorCode, dwFlags, dwParam, szUserStr)
{

    if(userType==2){//客服

        $("#Initiative_Call_Div").hide();//隐藏主动呼叫对话框

        $('#LOADING_GREY_DIV').hide();
```

进入房间成功后的逻辑代码如下：

```
// 客户端进入房间，dwRoomId表示所进入房间的ID号，errorcode表示是否进入房间：0成功进入，
// 否则为出错代码
function OnAnyChatEnterRoom(dwRoomId, errorcode) {
    if (errorcode == 0) {
        AddLog("function OnAnyChatEnterRoom(dwRoomId: "+dwRoomId+",errorcode: '"+errorcode,
LOG_TYPE_NORMAL);
        if (userType == 2) {

        } else if (userType == 1) {
            /**客服姓名*/
            var name1 = BRAC_GetUserInfo(mTargetUserId,USERINFO_NAME);
            /**个人姓名*/
            var myName = BRAC_GetUserInfo(mSelfUserId,USERINFO_NAME);

            var videoHtml='<div id="VideoShowDiv" style="display:block;margin-left:-80px;">'+
                '<div id="remoteVideoPos" class="videoshow0"></div>'+
                '<div id="remoteAudioVolume" style="width:480px;height:5px;top:370px;left:
21px;"></div>'+
                '<div id="localVideoPos" class="videoshow1"></div>'+
                '<div id="localAudioVolume" style="width:480px;height:5px;top:370px;left:
517px;"></div>'+
```

```
'<div id="div_username0" uid="" class="ShowName" style="left: 21px;">'+name1+'(客
服)</div>'+
'<div id="div_username1" uid="" class="ShowName" style="left:517px">'+myName+'(自
己)</div>'+
'<b style="position: absolute;bottom: -40;right: 30;font-size: 18px;"><a id="hangUp"
class="Buttons"></a></b>'+
'</div>';

$("#videoCallContent").html(videoHtml);//填充视频会话层
$("#LOADING_GREY_DIV").hide();
$("#videoCall").show();
$("#callLayer").hide();
}
//打开本地视频
startVideo(mSelfUserId, GetID("localVideoPos"), "ANYCHAT_VIDEO_LOCAL",1);
setVolumeTimer();//设置音量感应
} else {
    AddLog("function OnAnyChatEnterRoom(dwRoomId: "+dwRoomId+',errorcode: '+errorcode,
LOG_TYPE_ERROR);
}
}
```

进入房间成功后同时会打开双方的音视频以便进行视频交互。

## 7.4 视频呼叫结束

在整个视频通话服务完成后，任一方都可以通过调用 BRAC\_VideoCallControl 接口（视频呼叫控制接口）去发送 BRAC\_VIDEOCALL\_EVENT\_FINISH 事件结束当前的视频呼叫。参考代码如下：

```
//停止服务按钮事件(坐席)

$('#stopService').off().on('click', function () {

    if ($("#remoteVideoPos").html() != "") {

        if (confirm("你确定中止当前视频吗? ")) {

            BRAC_VideoCallControl(BRAC_VIDEOCALL_EVENT_FINISH,

mTargetUserId, 0, 0, 0, ""); // 挂断

            $("#Initiative_Call_Div").hide(); //隐藏主动呼叫层

        }

    }

});

// 绑定挂断按钮事件(客户)

$("#videoCallContent").delegate('#hangUp', 'click', function () {

    BRAC_VideoCallControl(BRAC_VIDEOCALL_EVENT_FINISH, mTargetUserId, 0,

0, 0, ""); // 挂断

    if (userType == 1) {

        $("#enterRoom h2").text(queueListName);

        /**离开队列*/
```

## 八、资源释放

### (1) 离开队列、营业厅

参考代码如下：

```
/**离开队列*/
    BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_QUEUE, queueid, ANYCHAT_QUEUE_CTRL_USERLEAVE, 0, 0, 0, 0, "");

/**离开营业厅*/
    BRAC_ObjectControl(ANYCHAT_OBJECT_TYPE_AREA, hallbusinessNum, ANYCHAT_AREA_CTRL_USERLEAVE, 0, 0, 0, 0, "");
```

在音视频交互结束后，可调用该操作。调用呼叫中心的 **BRAC\_VideoCallControl** 执行挂断即可。

### (2) 退出

断开与 AnyChat 通讯服务器连接。参考代码如下：

```
//退出，断开与服务器连接
BRAC_Logout();
```

在需要断开跟 AnyChat 服务器通讯连接的时候，可调用该操作。退出之后，可以再次调用连接、登录服务器。

### (3) 释放资源

释放整个 SDK 资源。刷新或者关闭网页就释放资源，不需要调用 api 释放。释放之后，需要重新调用初始化 SDK，才能进行连接、登录、进入房间等操作。



## 九、附录

本附录中包括了开发流程指南中所用到的示例程序的运行截图。

### 9.1 HelloAnyChat 界面

AnyChat for Web SDK 包里提供的 AnyChatWebDemo（源码在“src\AnyChatWebDemo”目录下）运行效果如下图所示：

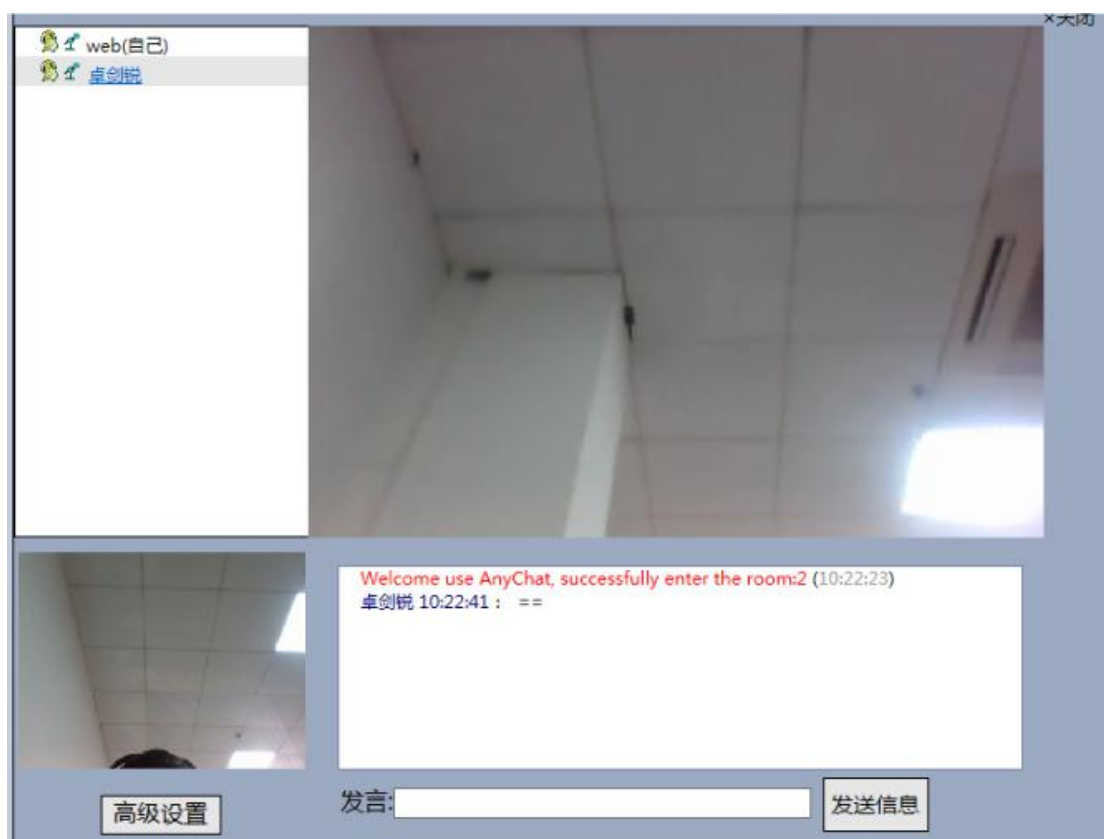


图 9-1 AnyChatWebDemo 主界面

### 9.2 AnyChatQueue 界面

AnyChat for Web SDK 包里提供的 AnyChatQueueDemo（源码在“src\AnyChatQueueDemo”目录下）运行效果如下图所示：



图 9-2 AnyChatQueue 登录页面

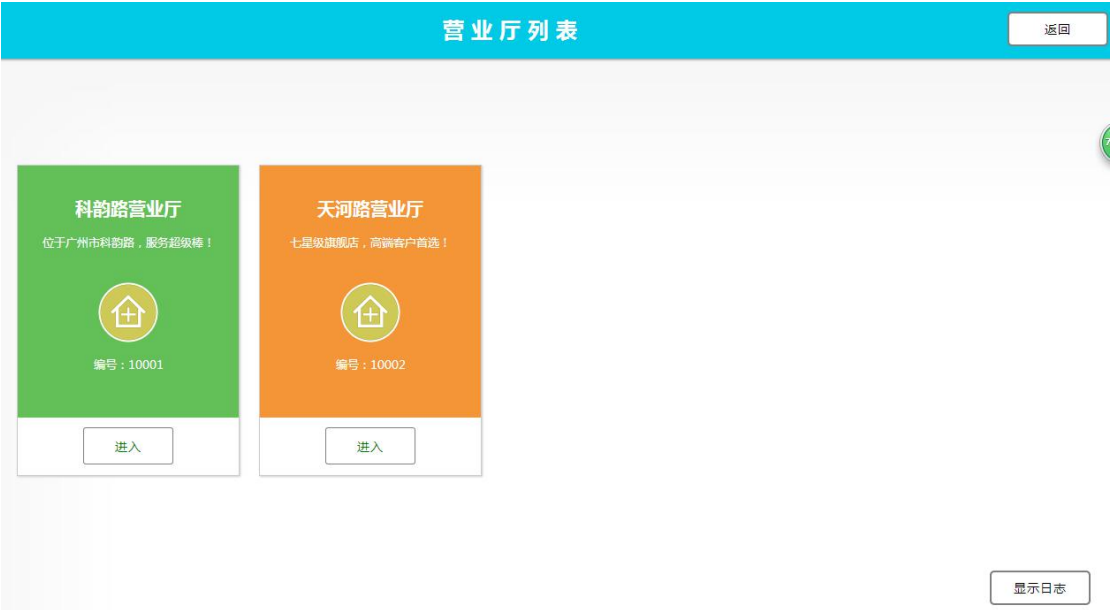


图 9-3 营业厅列表页面



图 9-4 营业厅内的队列列表页面



图 9-5 坐席(客服)端视频服务页面



图 9-6 客户端视频服务窗口页面