MVB2000 jMVBCTI 接口说明

部门 ： 产品部
版本/创建时间 ：1.0.4.0/2022-05-06

商标声明

MVB2000 和灵箭商标均为青岛畅信达通信有限公司的商标。

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、服务或特性等应受畅信达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，畅信达公司对本文档内容不做任何明示或默示的声明或保证。

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。

青岛畅信达通信有限公司	
地址：	青岛市市北区上清路 12 号中联 U 谷北 A2-506 室　　邮编：266022
网址：	http://www.ipxchina.cn
客户服务邮箱：	support@ipxchina.cn
客户服务电话：	4000-830-188
市场联络邮箱：	sales@ipxchina.cn
市场联络电话	4000-820-188

文档说明

文档主题： jMVBCTI Javascript API 编程手册

存放位置： SVN

编辑记录：

版本	日期	编写者	编辑原因
1.0.0.1	2012-07-01	畅信达	新建
1.0.0.3	2012-11-27	畅信达	增加： 满意度调查接口 发起呼叫接通指定应用接口 发起呼叫接通指定语音流程接口 建立会议多方通话接口 查询指定会议详情接口 会议锁定、踢出、禁言发言接口 获取坐席正在参加会议列表接口 获取坐席发起或参加会议列表接口 获取下载语音文件URL接口 获取平台URL接口 会议状态变化事件 弹屏对象增加队列号、原被叫、IVR历史、等待时间参数。
1.0.1.9	2013-11-05	畅信达	增加： UTF8 库 获取语音列表 IVR菜单列表 中继列表 Http请求错误事件
1.0.3.0	2014-10-27	畅信达	增加： 获取时间方法 闭音方法 取消闭音方法
1.0.3.1	2014-11-23	畅信达	增加：

			获取示忙原因方法 获取队列列表方法 修改示忙方法，增加一个示忙原因参数
V1.0.3.2	2014-12-02	畅信达	增加： GetQueueGeneral 获取队列基本数据方法
V1.0.3.3	2015-01-20	畅信达	修正文档标题
V1.0.3.4	2016-03-03	畅信达	增加： jMVBCTI_Bar 工具条
V1.0.3.5	2019-11-01	畅信达	修正文档格式，补充部分文档说明
V1.0.3.6	2020-05-28	畅信达	修正文档格式，补充部分文档说明
V1.0.3.7	2021-09-09	畅信达	公司信息变更
V1.0.4.0	2022-05-06	畅信达	SetAgent不再支持多次调用

目录

1 编写目的

本文所描述的轻量级第三方开发接口主要针对基于 B/S 架构的 Web 开发商，不限定开发工具，可以是 asp、jsp、php、java、.net 等任何一种。开发接口的表现形式为 JavaScript 脚本。

本文主要描述轻量级的第三方开发接口以及规范，主要阅读人员为第三方 CRM、ERP、信息管理系统等开发商以及具备二次开发能力的编程人员、系统设计人员。

本文所描述的轻量级第三方开发接口是对 MVB2000 平台 WEBCTI 接口的二次封装，适合大多数用户快速完成集成开发。有更多高级接口需求的用户请参考“MVB2000 接口类型”

2 概述

MVB2000 智能融合通信平台产品自 2007 年上市以来，已广泛应用于电信、政府、金融、税务、热电、电力、石油、学校、游戏产业、电子商务、高速公路、矿山、制造业等诸多行业，其稳定性、灵活性得到广大用户一致认可。MVB2000 平台提供了 SmartAgent、MVBCTI、WEBCTI、IAXOCX 等开发接口，用户可以使用这些接口实现专业的呼叫中心应用或与已有的 ERP、CRM、OA 等系统无缝对接。但由于接口的形式和使用方式对开发人员有一定要求，一些只需要实现简单呼叫中心功能的用户往往不需要了解很多深入的技术接口，他们希望使用几行代码就可以实现来电弹屏、点击拨号、转接、坐席上下线等基本呼叫中心功能，因此我们对 MVB2000 平台已有的接口进行了 2 次封装，以 javascript API 的形式提供给用户，供他们选择。需要作深入开发的用户仍然可以选择开放性更高、更灵活的接口。

3 MVB2000 主要接口类型

- MVBCTIOCX　一种基于微软 com 标准的 ActiveX 控件，包含 CTI、IVR、短信、传真、数据库等模块接口，适合所有编程语言。
- WEBCTI　采用 http 协议，通过 json 或 xml 进行数据交互的 CTI 接口。
- IAXOCX　一种基于微软 com 标准的 ActiveX 控件，提供软电话功能，程序员绘制一个程序界面就可以开发出专业的软电话或将电话功能嵌入到自己的应用程序中。
- SmartAgent　使用 MVBCTIOCX 和 IAXOCX 控件完成的坐席端软件，内嵌浏览器功能，提供呼叫中心常用按钮、会议、短信、传真管理。浏览器代码可以调用工具条任意按钮，如拨号、转接、上下线等。来电已事件方式通知浏览器。
- jMVBCTI　对 WEBCTI 进行 2 次封装的 javascript 接口，调用更简单，大多数功能 1 行代码实现，来电、坐席状态等全部以事件方式触发。
- 数据库接口　提供批量自动外呼、任务调度、高级短信、传真应用支持。

4 基本概念

- 用户：待分配的电话号码。
- 设备：指物理存在的终端设备。通过电话线或网络连接到 mvb2000 平台。
- 分机：通常指分配了号码的终端设备。是用户和设备的结合。
- 绑定：为设备指定一个关联的用户（号码）。只有类型为“临时”的设备允许此操作。
- 分离：取消与设备关联的用户。是绑定的反操作。只有类型为“临时”的设备允许此操作。
- 拨号接口：指明与终端进行通信的协议和编号。如:SIP/7001,IAX2/6001,ZAP/2。“/”前的是协议类型，后面的是唯一编号。
- 活动通道：活动通道用于唯一标识参与通话的通道。每路通话由 2 个活动通道组成，每个终端设备允许发起多路通话，每路通话都有自己独立的活动通道。转接、拆线等操作都以“活动通道”作为目标。如： IAX2/fax2-1807、SIP/627003-b7265460、SIP/192.168.1.8-b77044b8 等。
- 坐席：登录到队列上的设备。通常用设备的拨号接口标识。如 SIP/7001,IAX2/6001 等。
- 工号：与“用户”相同，登录到队列时设备所绑定的用户号作为坐席工作编号使用。
- 队列：用于自动分配呼叫、并提供等待排队功能的模块。以队列编号标识。每个坐席可以登录多个队列。队列可以包含固定坐席（开机就存在，不能登出）和动态坐席。
- 会议：实现多方同时进行语音互动、并提供发言管理功能的模块。以会议编号标识，分动态建立和固定号码两种。

5 配置设备与用户关系

- MVB2000 默认配置将设备与用户设置为固定关联关系，因此不允许动态绑定和分离。
- 如果您的每个坐席工位都只有一个固定的坐席使用，则不需要进行设备与用户设置为分离操作，也不需要进行动态绑定和分离。只需要操作登入和登出队列即可。
- 如果您的坐席工位由多个坐席轮流使用，需要修改配置，将设备与用户设置为分离状态，然后使用接口进行动态绑定和分离。
- 如何设置设备与用户分离：
  在 MVB2000 设置--》基本设置界面：

  选择“设备与分机分离”，提交后，刷新配置。在左边的菜单列表中会出现设备和用户。
- 使用分离方式时，用户号和设备号最好不要重复。比如设备号使用 5000 号段，用户号使用 6000 号段。
- 使用分离方式时，未绑定用户号码的设备只能呼出不能呼入。

6 认证密钥

6.1 设置 WEB 密钥

首先在 MVB2000 设置--》基本设置界面设置您需要的 WEB 密钥，如：

，然后提交。您也可以不修改原有值，直接提交。此密钥为动态生成，每台设备并不相同。只有点击过“提交”按钮后，web 密钥才可以使用。系统第一次安装或使用时，也需要击过“提交”按钮。

6.2 计算密钥参数

webkey 使用 mvb2000 基本设置里的 webkey 字符串+当日日期（格式：yyyy-mm-dd）组成的字符串进行 MD5 运算，得到的结果作为 webkey 使用。结果应该全部以 ascii 码方式表示，使用**小写**字母。计算结果应该是 32 位的字符串，也随日期变化。您可以参照 demo 代码中的 js 脚本，将此功能编写为函数。或者使用后台语言编写（php/c#/java 等）。不同编程语言返回的字节顺序可能不同，请参考以下结果调整：

“123456”通过 MD5 加密后结果为：e10adc3949ba59abbe56e057f20f883e

由于认证密钥生成时使用了日期要素，请确保：

1、 客户端日期与平台日期一致

2、 日期变化后要重新计算认证密钥，并使用 SetWebKey 更新。

6.3 测试密钥参数

一、首先确认 mvb2000 平台系统日期准确（可使用 mvb2000 设置→系统设置→系统时间设置修改）

二、确保 http 或 https 可以访问（MVB2000 平台默认未开启 http，您可以使用 https 访问 mvb2000 设置→系统设置→WEB 服务设置开启 http）

三、打开浏览器，在地址栏输入：

http://192.168.0.218/webservice/getcpuload.php?webkey=5589a16d0b48a49ce1c66c2ad11268c0

7 JMVBCTI 引入方法

- **引入 jquery 库**

JMVBCTI 使用 jquery 库的跨域解决方案，支持 v1.3.2 或以上版本，如果您的应用也使用 jquery，则可以不包含平台的 jquery 库。

以下版本已经过测试且随 webcti-1.0.1.9 安装包安装:

jquery-1.3.2.min.js jquery-1.4.4.min.js jquery-1.6.4.min.js jquery-1.7.2.min.js jquery-1.8.3.min.js jquery-1.9.1.min.js jquery-1.10.2.min.js jquery-2.0.3.min.js(只支持 IE9 以上)

引入方法:

```
<script type="text/javascript"
src="http://192.168.0.218/webservice/jmvbcti/jquery-1.10.2.min.js" ></script>
```

- 引入 jMVBCTI 库

```
<script type="text/javascript"
src="http://192.168.0.218/webservice/jmvbcti/jMVBCTI.js" ></script>
```

或 UTF8 版本

```
<script type="text/javascript"
src="http://192.168.0.218/webservice/jmvbcti/jMVBCTI_UTF8.js" ></script>
```

- 接口库名称

JMVBCTI 接口的库名称：jMVBCTI

使用该接口的方法和属性都采用 jMVBCTI.methd 的形式，如拨号：jMVBCTI.dial(‘4000830188’)

8 函数

8.1 About 显示接口版本号

函数原型	About ()			
功能	显示接口版本号。			
示例	jMVBCTI. About ( );			
参数	名称	类型	用途	参数实例
	无			
返回值	无			

8.2 Bind 绑定工号与设备

函数原型	Bind (parm, callback)			
功能	绑定工号与设备。			
示例	jMVBCTI. Bind ( ); jMVBCTI. Bind ({},callback); jMVBCTI. Bind ({agentno: ' 7001' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，可选	
parm 对象成员	callback	function	回调函数，可选	
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息,需要 decodeURI	resdata. message

8.3 CallApplication 发起呼叫接通指定应用

函数原型	CallApplication (parm, callback)			
功能	发起呼叫接通指定应用接口 先呼叫 channel 指定的目标，目标应答后执行 application 指定的语音应用。			
示例	var parm={channel: ' SIP/7001 ' ,application:'Playback',data: ' beep' ,async:'0' }; jMVBCTI. ConfCreate (parm ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	channel	string	拨号接口 必填	SIP/7001 Zap/g0/82754888

				Local/82754888@context
	agentno	string	工号，可选 默认为调用 SetAgent 加入的坐席	7001
	callee	string	话单被叫号码 可选	82754888
	callerid	string	主叫号码,呼叫参会者时发送的来显号码(数字中继线上需要送合规的号码），可选	7001
	application	string	应用名称,必选	Playback MeetMe Dial DEADAGI AGI …
	data	string	data 应用参数 可选	SIP/102100-341ac
	timeout	string	拨号超时(毫秒),等待被叫接通的时间。 默认：45000。 可选	30000
	variable	string	预设通道变量 可选。 用来与交换部分交互。	
	account	string	计费账号，记录到话单的账号。可选	7001
	async	string	是否异步(0/1) 默认 0-同步 异步：发送指令立即返回 同步：发送指令等待接通 可选	0
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.reason	string	呼叫失败原因 详见《发起呼叫返回 Reason 说明》	3

8.4 CallContext 发起呼叫接通指定语音流程

函数原型	CallApplication (parm, callback)			
功能	发起呼叫接通指定语音流程接口 先呼叫 channel 指定的目标，目标应答后执行 context 指定的语音流程。			
示例	var parm={channel:’SIP/7001’,context:’ivr-2’,exten:’s’,async:’0’}; jMVBCTI. ConfCreate (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	channel	string	拨号接口 必填	SIP/7001 Zap/g0/82754888

				Local/82754888@context
	agentno	string	工号，可选 默认为调用 SetAgent 加入的坐席	7001
	callee	string	话单被叫号码 可选	82754888
	callerid	string	主叫号码,呼叫参会者时发送的来显号码(数字中继线上需要送合规的号码），可选	7001
	context	string	语音流程入口 必填	ivr-2
	exten	string	语音流程号码 必填	s
	timeout	string	拨号超时(毫秒),等待被叫接通的时间。 默认：45000。 可选	30000
	variable	string	预设通道变量 可选。 用来与交换部分交互。	
	account	string	计费账号，记录到话单的账号。可选	7001
	async	string	是否异步(0/1) 默认 0-同步 异步：发送指令立即返回 同步：发送指令等待接通 可选	0
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.reason	string	呼叫失败原因 详见《发起呼叫返回 Reason 说明》	3

8.5 ConfCreate 建立会议（多方通话）

函数原型	ConfCreate (parm, callback)			
功能	建立动态/静态会议(多方通话)接口			
示例	var parm={confno: ’ 95678 ’ ,conftype:’D’,phonenolist: ’ 82754888#|8001’ ,async:’0’,activechannel1:activechannel1,activechannel2:activechannel2}; jMVBCTI. ConfCreate (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	confno	string	会议室号码(可以使用 5-8 位数字或已有的静态会议室号码）可选。 如不提供此参数，接口随机产生会议室号码（900000-	98100

			999999），并在 json 对象返回。	
	agentno	string	工号，可选 默认为调用 SetAgent 加入的坐席	7001
	conftype	string	会议类型(D/S D-动态 S-静态) 可选，默认动态	D
	phonenolist	string	邀请参会的电话号码列表，采用|分隔。外线号码用#后缀。 必填	8001|82754888#|8002
	callerid	string	主叫号码，呼叫参会者时发送的来显号码（数字中继线上需要送合规的号码），可选	7001
	activechannel1	string	正在进行的一组通话的活动通道，必须同时提供 2 个通道。 可选	SIP/7001-12034ab
	activechannel2	string		SIP/102100-341ac
	timeout	string	拨号超时(毫秒),等待被叫接通的时间。 可选	30000
	variable	string	预设通道变量 可选。 用来与交换部分交互。会议建立的固定参数是：dqM 。如果想控制会议室的其他行为，可以设置变量“MVB_NWAYOPTION”，如：“MVB_NWAYOPTION=P,1234”要求输入密码 1234。	MVB_NWAYOPTION=P,1234
	account	string	计费账号，记录到话单的账号。 可选	7001
	async	string	是否异步(0/1) 默认 0-同步 异步：发送指令立即返回 同步：发送指令等待接通 可选	0
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.confno	string	动态会议室号码	98100
	data.sucnums	string	呼叫成功号码数量	3
	data.redirect	string	将活动通道转入会议是否成功。Failed/Success	Success
	data.redirectmessage	string	活动通道转入会议失败原因	Success

	data.reason_ 号码 1 data.reason_ 号码 2 …	string	呼叫该号码失败原因，成功时为空串	
	data.号码 1 data.号码 2 …	string	呼叫该号码结果。Failed/Success	Success

8.6 ConfDetail 获取会议详情

函数原型	ConfDetail (parm, callback)			
功能	获取会议详情接口			
示例	var parm={confno:’95678’ }; jMVBCTI. ConfCreate (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	confno	string	会议室号码，必选。	98100
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data	json 对象	存放通话列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	createtime	string	进入会议室时间	2012-11-15 15:45:12
	channel	string	活动通道	SIP/7001-7865de
	callerid	string	与会者电话号码	82754777
	uniqueid	string	呼叫编号	114587995.369
	usernum	string	与会者编号	1

8.7 ConfExitAll 坐席退出参与的所有会议

函数原型	ConfExitAll (parm, callback)			
功能	坐席退出参与的所有会议接口			
示例	var parm={agentno:’7001’, exitall:’0’ }; jMVBCTI. ConfExitAll (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	agentno	string	坐席号码，可选。	7001
	exitall	string	是否关闭会议(0/1) 0-退出 1-退出并关闭会议	0

回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	名称	类型	用途	参数实例
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.confno	string	退出的会议室号码	98001
	data.usernum	string	退出的与会者编号	all 1

8.8 ConfGet 获取坐席正在参与的会议对象

函数原型	ConfGet (parm, callback)			
功能	获取坐席正在参与的会议对象接口，与 GetConfNos 不同，此方法只返回坐席当前参加的会议列表。 此方法从平台获取数据。			
示例	var parm={agentno:’7001’ }; jMVBCTI. ConfGet(parm ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	agentno	string	坐席号码，可选。	7001
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	名称	类型	用途	参数实例
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data	json 对象	存放通话列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	confno	string	会议室号码	98001
	usernum	string	与会者编号	1

8.9 ConfKick 踢出指定与会者

函数原型	ConfKick (parm, callback)			
功能	踢出指定与会者接口			
示例	var parm={ confno:’98001’,usernum:’1’ }; jMVBCTI. ConfKick (parm ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	confno	string	会议号码，必选。	98001
	usernum	string	与会者编号，可选。不填则踢	1

			出全部与会者，等同于填”all”。	all
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0代表成功	resdata. errcode
	message	string	错误信息，需要decodeURI	resdata. message
	data.confno	string	退出的会议室号码	98001
	data.usernum	string	退出的与会者编号	all 1

8.10 ConfLock 锁定会议室

函数原型	ConfLock (parm, callback)			
功能	锁定会议室接口，不允许其它成员加入			
示例	var parm={ confno:’98001’ }; jMVBCTI. ConfLock (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json对象	传递参数，必选	
	callback	function	回调函数，可选	
parm对象成员	confno	string	会议号码，必选。	98001
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0代表成功	resdata. errcode
	message	string	错误信息，需要decodeURI	resdata. message
	data.confno	string	锁定的会议室号码	98001

8.11 ConfUnlock 解锁会议室

函数原型	ConfUnlock (parm, callback)			
功能	解锁会议室接口，允许其它成员加入。			
示例	var parm={ confno:’98001’ }; jMVBCTI. ConfUnlock (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json对象	传递参数，必选	
	callback	function	回调函数，可选	
parm对象成员	confno	string	会议号码，必选。	98001
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0代表成功	resdata. errcode
	message	string	错误信息，需要decodeURI	resdata. message

	data.confno	string	允许的会议室号码	98001

8.12 ConfMute 禁止与会者发言

函数原型	ConfMute (parm, callback)			
功能	禁止与会者发言接口			
示例	var parm={ confno:'98001',usernum:'1' }; jMVBCTI. ConfMute (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	confno	string	会议号码，必选。	98001
	usernum	string	与会者编号，可选。不填则禁言全部与会者，等同于填"all"。	1 all
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.confno	string	禁言的会议室号码	98001
	data.usernum	string	禁言的与会者编号	all 1

8.13 ConfUnmute 允许与会者发言

函数原型	ConfUnmute (parm, callback)			
功能	允许与会者发言接口			
示例	var parm={ confno:'98001',usernum:'1' }; jMVBCTI. ConfUnmute (parm ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必选	
	callback	function	回调函数，可选	
parm 对象成员	confno	string	会议号码，必选。	98001
	usernum	string	与会者编号，可选。不填则允许全部与会者，等同于填"all"。	1 all
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data.confno	string	允许的会议室号	98001

			码	
	data.usernum	string	允许的与会者编号	all 1

8.14 CallWebCTI 调用 WEBCTI 高级接口

函数原型	CallWebCTI (parm, callback)			
功能	调用 WEBCTI2 高级接口			
示例	jMVBCTI. CallWebCTI ('getarea.php' ,{phoneno: '189063979XX'} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	ctiaction	string	接口名称，必选	getarea.php
	parm	json 对象	传递参数，必选	{phoneno:189063979XX}
	callback	function	回调函数，可选	
parm 对象成员	详见《MVB2000_WEBCTI2 开发接口》各接口参数。			
回调函数原型	callback (jsondata)也可以不指定回调函数			
回调函数参数	详见《MVB2000_WEBCTI2 开发接口》各接口返回值。			

8.15 Dial 拨号/外呼

函数原型	Dial (parm, callback)			
功能	拨打指定电话，采用先呼坐席模式，坐席摘机后才会拨打指定号码。			
示例	jMVBCTI. Dial ({phoneno :' 4000820188' }); jMVBCTI. Dial ({phoneno :' 4000820188' ,agentno :' 7001' }, callback); jMVBCTI. Dial ({phoneno :' 4000820188' ,agentno :' 7001' , callerid:' 7001}, callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，可选	
parm 对象成员	phoneno	string	电话号码	4000820188
	agentno	string	工号，可选 默认为调用 SetAgent 加入的坐席	7001
	callerid	string	主叫号码，可选	7001
	async	string	是否异步，可选 0- 同步 1- 异步 默认 0	0
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.

				message

8.16 GetConfNos 获取坐席发起或参与的会议列表

函数原型	GetConfNos (agentno)			
功能	获取坐席发起或参与的会议列表，返回 confnos 对象。与 ConfGet 不同，此方法返回坐席当前参加的和坐席发起但已经退出的会议列表。 此方法从本地坐席对象获取数据。			
示例	jMVBCTI. GetConfNos （‘7001’）			
参数	**名称**	**类型**	**用途**	**参数实例**
	agentno	string	坐席号码，必选。	7001
返回结果	**名称**	**类型**	**用途**	**参数实例**
	data	json 对象	存放用户列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	**成员**	**类型**	**用途**	**参数实例**
	confno	string	会议室号码	confnos [0].confno: 998845
	usernum	string	坐席在会议室中与会者编号,坐席退出后,此编号为空字符串。	confnos [0].usernum:2

8.17 GetMVBServerURL 返回平台 URL 地址

函数原型	GetMVBServerURL ()	
功能	返回平台 URL 地址 参数:无 已结果集方式返回 URL 地址。	
示例	jMVBCTI. GetMVBServerURL ()	
参数	**无**	
返回	url 地址字符串	如：http://192.168.0.218

8.18 GetNotifyg 获取消息提示开关

函数原型	GetNotify()			
功能	获取消息提示开关状态。			
示例	jMVBCTI. GetNotify ();			
参数	**名称**	**类型**	**用途**	**参数实例**
	无			
返回值	B	布尔值	开关(true/false)	true

8.19 GetDebug 取调试级别设置

函数原型	GetDebug ()			
功能	取调试级别设置。			
示例	jMVBCTI. GetDebug ();			
参数	名称	类型	用途	参数实例
	无			
返回值	level	数值	调试级别（0-3）	1

8.20 GetAllCalls 获取所有通话列表

函数原型	GetAllCalls (callback)			
功能	获取当前所有通话列表，以回调函数方式返回			
示例	jMVBCTI. GetAllCalls (callback)			
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放通话列表，根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	createtime	string	数据生成时间	resdata.data[i]. createtime 2012-07-02 14:09:06
	srcchannel	string	源活动通道	resdata.data[i].srcchannel SIP/7001-0a015460
	dstchannel	string	目标活动通道	resdata.data[i].dstchannel SIP/100-102-0a005df0
	callstate	string	呼叫状态 Dialing/Link	resdata.data[i]. callstate Link
	srccallerid	string	源通道主叫号	resdata.data[i]. srccallerid 7001
	srccalleridname	string	源通道主叫名 需要 decodeURI 解码	resdata.data[i]. srccalleridname <unknown>
	dstcallerid	string	目标通道主叫号	resdata.data[i]. dstcallerid 8001
	dstcalleridname	string	目标通道主叫名 需要 decodeURI 解码	resdata.data[i]. dstcalleridname <unknown>
	srcchannelstate	string	源通道状态	resdata.data[i]. srcchannelstate

	dstchannelstate	string	目标通道状态	resdata.data[i].dstchannelstate
	srcuniqueid	string	源通道呼叫编号	resdata.data[i].srcuniqueid 1341209319.2894
	dstuniqueid	string	目标通道呼叫编号	resdata.data[i].dstuniqueid 1341209319.2895
	dialtime	string	拨号时间	resdata.data[i]. dialtime 2012-07-02 14:08:39
	linktime	string	接通时间	resdata.data[i]. linktime 2012-07-02 14:09:06
返回值	无			

8.21 GetAllUsers 获取所有用户(号码)列表

函数原型	GetAllUsers (callback)			
功能	获取平台所有用户列表，以回调函数方式返回			
示例	jMVBCTI. GetAllUsers (callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放用户列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	user	string	号码(用户)	resdata.data[i]. user 7001
	name	string	名称 需要 decodeURI 解码	resdata.data[i].name default
返回值	无			

8.22 GetAllDevice 获取所有设备列表

函数原型	GetAllDevices (callback)			
功能	获取平台所有设备列表，以回调函数方式返回			
示例	jMVBCTI. GetAllDevices (callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放设备列表,根据	resdata.data[i].varname

			resdata.data.length 循环历遍每条记录	resdata.data[i].length
data 对象集合	device	string	设备号码	resdata.data[i]. device 7001
	tech	string	协议类型 sip/iax2/zap/custom	resdata.data[i]. tech sip
	dial	string	接口	resdata.data[i]. dial SIP/7001
	devicetype	string	绑定类型 adhoc/fixed	resdata.data[i]. devicetype fixed
	context	string	拨号授权	resdata.data[i]. context from-internal-1
返回值	无			

8.23 GetAllQueues 获取所有队列列表

函数原型	GetAllQueues (callback)			
功能	获取平台全部队列列表，以回调函数方式返回			
示例	jMVBCTI. GetAllQueues (callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放队列列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	queueno	string	队列号码	resdata.data[i]. queueno 7910
返回值	无			

8.24 GetTrunkList 获取所有中继列表

函数原型	GetTrunkList(parm,callback)			
功能	获取平台全部中继列表，以回调函数方式返回			
示例	jMVBCTI. GetTrunkList({},callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放中继列表,根据 resdata.data.length	resdata.data[i].varname resdata.data[i].length

			h 循环历遍每条记录	
data 对象集合	trunk	string	中继名称	resdata.data[i]. trunk Zap/g0
返回值	无			

8.25 GetIVRList 获取所有 IVR 列表

函数原型	GetIVRList(parm,callback)			
功能	获取平台全部 IVR 列表，以回调函数方式返回			
示例	jMVBCTI. GetIVRList({},callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放列表，根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	id	string	IVR 的 ID	resdata.data[i]. id Ivr-2
	ivrname	string	IVR 名称 需 decodeURI 解码	resdata.data[i]. ivrname welcomemenu
返回值	无			

8.26 GetPromptFiles 获取所有自定义语音列表

函数原型	GetPromptFiles(parm,callback)			
功能	获取平台全部自定义语音列表，以回调函数方式返回			
示例	jMVBCTI. GetPromptFiles({},callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放队列列表，根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	id	string	IVR 的 ID	resdata.data[i]. id 90
	filename	string	文件名称	resdata.data[i]. filename welcome
	description	string	语音描述 需 decodeURI 解码	resdata.data[i]. description 欢迎致电畅信达
返回值	无			

8.27 GetAgents 返回坐席对象集合

函数原型	GetAgents ()
功能	获取坐席对象集合，以 json 对象方式返回
示例	var agentlist=jMVBCTI. GetAgents ()
返回值	坐席对象集合 Agents,每行一个对象,使用 Agents.length, Agents[i]. varname 方式访问 详见数据对象部分。

8.28 GetQueueCalls 获取当前排队列表

函数原型	GetQueueCalls (parm,callback)			
功能	获取当前排队列表，以回调函数方式返回			
示例	jMVBCTI. GetQueueCalls ({},callback)			
参数	**名称**	**类型**	**用途**	**参照方法**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，必须	
parm 对象成员	**名称**	**类型**	**用途**	**参照方法**
	queueno	string	队列号，可选	{queueno:’ 7910’ }
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json 对象	存放排队列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].varname resdata.data[i].length
data 对象集合	createtime	string	数据生成时间	resdata.data[i]. createtime 2012-07-02 14:09:06
	queue	string	队列号	resdata.data[i].queue 7910
	queuestart	string	时戳	resdata.data[i]. queuestart 1341212843
	channel	string	活动通道	resdata.data[i]. channel SIP/100-102-0a005df0
	uniqueid	string	呼叫编号	resdata.data[i]. uniqueid 1341209319.2894
	callerid	string	主叫号码	resdata.data[i]. callerid 8001
	calleridname	string	主叫名 需要 decodeURI 解码	resdata.data[i]. calleridname <unknown>
	position	string	当前位置	resdata.data[i]. position 1
	wait	string	等待秒数	resdata.data[i]. wait

				45
	count	string	次数	resdata.data[i]. count 0
返回值	无			

8.29 GetCalls 返回坐席当前通话列表

函数原型	GetCalls (agentno)
功能	获取坐席当前通话列表，以 json 对象方式返回,接口只返回当前实例的通话列表。
示例	var calls=jMVBCTI. GetCalls ()
参数	agentno 坐席号码，默认为通过调用 SetAgent 登记的坐席号码，通常不需要指定。
返回值	坐席当前通话集合 calls，每行一个对象，使用 calls.length, calls [i].varname 方式访问 详见数据对象部分。

8.30 GetQueueGeneral 获取队列概况

函数原型	GetQueueGeneral(queuenolist,callback)			
功能	获取获取队列概况，以 json 对象方式返回。			
示例	jMVBCTI.GetQueueGeneral(queuenolist,callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json对象	存放队列概况列表,根据 resdata.data.length 循环历遍每条记录	resdata.data[i].queue resdata.data[i].length
data对象集合	queue	字符串	队列号码	resdata.data[i]. queue 90
	membercount	整型	坐席数	resdata.data[i].membercount 5
	waitings	整型	排队数	resdata.data[i].waitings 2
	usable	整型	示闲并且设备可用数	resdata.data[i].usable 3
	unusable	整型	示忙或设备状态不正常数	resdata.data[i].unusable 2
	inuse	整型	正在通话或振铃数	resdata.data[i].inuse 1
	usableidle	整型	示闲并且设备可用并空闲数	resdata.data[i]. usableidle 2
	pause	整型	示忙数	resdata.data[i].pause 1
	unpause	整型	示闲数	resdata.data[i].Unpause 4
	invalid	整型	设备状态不正常数	resdata.data[i].invalid 1
返回值	无			

8.31 GetParks 返回呼叫保持对象

函数原型	GetParks (agentno)
功能	获取坐席队列对象，以 json 对象方式返回。通过此函数可以取得坐席呼叫保持数据。
示例	var queues =jMVBCTI. GetParks ()
参数	agentno 坐席号码，默认为通过调用 SetAgent 登记的坐席号码，通常不需要指定。
返回值	呼叫保持数据集合 parks，每行一个对象，使用 parks.length, parks [i]. varname 方式访问 详见数据对象部分。

8.32 GetPauseTypeList 返回示忙原因列表

函数原型	GetPauseTypeList({},callback)			
功能	获取平台示忙原因列表，以回调函数方式返回			
示例	jMVBCTI. GetPauseTypeList({},callback)			
回调函数原型	callback (resdata)			
回调函数参数	**名称**	**类型**	**用途**	**参照方法**
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json对象	存放列表，根据 resdata.data.length 循环历遍每条记录	resdata.data[i].pausetype resdata.data[i].length
data 对象集合	pausetype	string	编号(1-99)	resdata.data[i].pausetype
	description	string	原因(示忙/培训等) 需 decodeURI 解码	resdata.data[i].description
返回值	无			

8.33 GetPopUpCalls 返回坐席弹屏数据列表

函数原型	GetPopUpCalls (agentno)
功能	获取坐席弹屏数据列表，以 json 对象方式返回。接口只会返回当前实例的弹屏数据列表。
示例	var popupcalls=jMVBCTI. GetPopUpCalls ()
参数	agentno 坐席号码，默认为通过调用 SetAgent 登记的坐席号码，通常不需要指定。
返回值	坐席当前弹屏数据集合 popupcalls，每行一个对象，使用 popupcalls.length, popupcalls [i]. varname 方式访问 详见数据对象部分。

8.34 GetQueueList 获取队列号码和名称列表

函数原型	GetQueueList(parm,callback)			
功能	获取平台队列列表，以回调函数方式返回			
示例	jMVBCTI. GetQueueList({},callback)			
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息	resdata. message
	data	json对象	存放队列列表,根据resdata.data.length循环历遍每条记录	resdata.data[i].queueno resdata.data[i].length
data对象集合	id	string	号码	resdata.data[i]. queueno
	ivrname	string	描述	resdata.data[i].description
返回值	无			

8.35 GetQueues 返回坐席队列对象

函数原型	GetQueues (agentno)
功能	获取坐席队列对象，以 json 对象方式返回。通过此函数可以取得坐席在每个队列(SetQueue 指定)的上线情况。
示例	var queues =jMVBCTI. GetQueues ()
参数	agentno 坐席号码，默认为通过调用 SetAgent 登记的坐席号码，通常不需要指定。
返回值	坐席当前弹屏数据集合 queues，每行一个对象，使用 queues.length, queues[i]. varname 方式访问 详见数据对象部分。

8.36 GetDeviceStatus 返回坐席线路状态

函数原型	GetDeviceStatus (agentno)
功能	获取坐席线路状态。
示例	var queues =jMVBCTI. GetDeviceStatus ()
参数	agentno 坐席号码，默认为通过调用 SetAgent 登记的坐席号码，通常不需要指定。
返回值	数值，详见<附表.坐席线路状态>

8.37 GetQueueStatus 返回坐席状态

函数原型	GetQueueStatus (parm)
功能	获取坐席在队列的状态。
示例	var result =jMVBCTI. GetQueueStatus ({agentno :’7001’, queueno:’

	7910’ }); var result =jMVBCTI. GetQueueStatus ({queueno:’ 7910’ }); var result =jMVBCTI. GetQueueStatus ();			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必须	
parm 对象成员	agentno	string	工号，可选 默认为调用 SetAgent 加入的坐席	7001
	queueno	string	队列号，可选 默认第一个用 SetQueue 加入的队列	7910
返回值	result	json 对象	只有一行数据的 json 对象	
返回 json 对象成员	queueno	string	队列号	result. queueno 7910
	agentno	string	坐席号码	result. agentno 7001
	ismember	string	上线状态(0/1)	result. ismember 1
	agent	string	坐席接口	result. queueno SIP/7001
	agentship	string	坐席类型(static/dynamic)	result. agentship dynamic
	penalty	string	坐席优先级	result. queueno 0
	paused	string	示忙状态(0/1)	result. paused 1

8.38 GetVars 批量获取通道变量

函数原型	GetVars (parm, callback)			
功能	批量获取通道变量。			
示例	jMVBCTI. GetVars ({channel: ’ SIP/100-102-0a005df0’, variable0: ’ DID’ } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，必须	
parm 对象成员	channel	string	活动通道 ， 必须	SIP/100-102-0a005df0
	variable0-variable9	string	变量名 0-9，至少一项	DID
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode

	message	string	错误信息，需要 decodeURI	resdata. message
	data	json 对象	存放返回变量	resdata.data.varname
data 对象集合	variable0-9	string	变量名 0-9 的值	resdata.data.variable0 resdata.data.variable9

8.39 GetChannelDetails 批量获取常用通道变量

函数原型	GetChannelDetails (parm, callback)			
功能	批量获取通道变量。			
示例	jMVBCTI. GetVars ({channel: ' SIP/100-102-0a005df0', variable0: 'DID'} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，必须	
parm 对象成员	channel	string	活动通道 ，必须	SIP/100-102-0a005df0
	timeout	string	超时秒数(如果未接通，等待接通时间) 可选	30
	gettype	string	获取方式 可选 0--返回当前通道变量 1--返回对方通道变量 2--自动判断(自动查找主叫通道)，通常主叫通道变量较为详尽和丰富 默认 0	2
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data	json 对象	存放返回变量	resdata.data.varname
data 对象集合	channel	string	通道	resdata.data. channel SIP/100-102-0a005df0
	uniqueid	string	编号	resdata.data. uniqueid
	dstchannel	string	目标通道	resdata.data. stchannel
	dstuniqueid	string	目标编号	resdata.data. dstuniqueid
	did	string	原被叫	resdata.data.did
	monitorfile	string	录音文件名称	resdata.data.monitorfile
	dialednum	string	已拨叫号码	resdata.data.dialednum

	queuenum	string	队列号	resdata.data.queuenum
	agentno	string	坐席号	resdata.data.agentno
	ivrhist	string	IVR 路径	resdata.data.ivrhist
	holdtime	string	等待时间	resdata.data.holdtime
	caller	string	主叫	resdata.data. caller 7001
	callee	string	被叫	resdata.data. callee 8001
	areacode	string	区号	resdata.data. areacode 0532
	province	string	归属省 需 decodeURI	resdata.data. province 山东
	city	string	归属市 需 decodeURI	resdata.data. city 青岛
	simcardtype	string	卡类型 需 decodeURI	resdata.data.simcardtype 联通卡

8.40 GetDateTime 获取平台日期时间

函数原型	GetDateTime(callback)				
功能	获取平台日期时间。				
示例	jMVBCTI.GetDateTime(callback);				
参数	**名称**	**类型**	**用途**	**参数实例**	
	callback	function	回调函数，可选		
回调函数原型	callback (resdata)				
回调函数参数	名称	类型	用途	参照方法	
	errcode	number	错误代码，0 代表成功	resdata.errcode	
	message	string	错误信息，需要 decodeURI	resdata.message	
	data	json 对象	存放返回变量	resdata.data.varname	
data 对象集合	mvb_datetime	string	日期时间		
	mvb_date	string	日期		
	mvb_time	string	时间		
	mvb_timestamp	number	时间戳		
	mvb_year	number	4 位年		
	mvb_month	number	月(1-12)		
	mvb_month_day	number	日(1-31)		
	mvb_year_day	number	一年第几天(1-366)		
	mvb_week	number	星期(0-6)		
	mvb_hour	number	时		
	mvb_minute	number	分		
	mvb_second	number	秒		

8.41 Hold 坐席闭音

函数原型	Hold (parm,callback)			
功能	坐席闭音，分机静音，对方听音乐			
示例	jMVBCTI. Hold ( ); jMVBCTI. Hold ({},callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	callback	function	回调函数，可选	
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息,需要 decodeURI	resdata. message

8.42 HangUp 拆线

函数原型	HangUp (parm, callback)			
功能	拆线。			
示例	jMVBCTI. HangUp ( ); jMVBCTI. HangUp ({callindex: 1} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	callindex	number	呼叫索引 可选。用以区分坐席进行的多路活动通话。 默认:0	0
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息,需要 decodeURI	resdata. message

8.43 Init 接口初始化

原型	Init(mvbserverURL)			
功能	初始化接口库，必须在所有函数使用前调用，是使用接口的第一步。			
示例	jMVBCTI.Init("http://192.168.0.102");			
参数	**名称**	**类型**	**用途**	**参数实例**

	mvbserverURL	string	平台 URL	http://192.168.0.218
返回值	无			

8.44 OnLine 坐席上线

函数原型	OnLine (parm, callback)			
功能	坐席上线。			
示例	jMVBCTI. OnLine ( ); jMVBCTI. OnLine ({},callback); jMVBCTI. OnLine ({queueno: ' 7910' } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	queueno	string	队列号码 ， 可选	7910
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.45 OffLine 坐席下线

函数原型	OffLine (parm, callback)			
功能	坐席下线。			
示例	jMVBCTI. OffLine ( ); jMVBCTI. OffLine ({},callback); jMVBCTI. OffLine ({queueno: ' 7910' } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	queueno	string	队列号码 ， 可选	7910
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.46 Park 呼叫保持（驻留）

函数原型	Park (parm, callback)			
功能	保持通话,保持后话机释放。			
示例	jMVBCTI. Park (); jMVBCTI. Park ({timeout: ' 300' } ,callback); jMVBCTI. Park ({agentno: '7001' ,timeout: ' 300' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	timeout	string	保持时长，可选 默认 300 秒	300
	callindex	number	呼叫索引 可选。用以区分坐席进行的多路活动通话。 默认:0	0
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.47 Pause 坐席示忙

函数原型	Pause (parm, callback)			
功能	坐席示忙。			
示例	jMVBCTI. Pause ( ); jMVBCTI. Pause ({},callback); jMVBCTI. Pause ({queueno: '7910' } ,callback); jMVBCTI. Pause ({queueno: '7910' ,pausetype:' 2' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	queueno	string	队列号码 ， 可选	7910
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
	pausetype	string	示忙原因	1-99 的数字编号 默认 1，每个编号的含义在平台队列基本参数

				中定义。
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.48 PickUp 代接/抢接通话

函数原型	PickUp (parm, callback)			
功能	代接/抢接通话。			
示例	jMVBCTI. PickUp ({pickupexten: ' 7002' }); jMVBCTI. PickUp ({pickupexten: ' 7002' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	pickupexten	string	号码 ,必须	7002
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.49 QM 满意度调查（质检）

函数原型	QM (parm, callback)			
功能	满意度调查(质检)接口			
示例	jMVBCTI. QM (); jMVBCTI. QM ({queuename: ' 7910' , agentno:' 7001' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	queuename	string	队列号码，可选 不提供时从活动通道的通道变量”MVB_QUEUENUM”获取	7910
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
	callindex	int	活动通话索引,可选	0

	ownerapp	string	应用系统标记,可选	csr2000
	sound1	string	提示语音,可选	custom/welcome
	sound2	string	感谢语音,可选	custom/thanks
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.50 SetDebug 设置调试级别

原型	SetDebug(level)			
功能	设置调试级别，需在 Start 前调用。控制接口是否以 alert 方式输出错误和调试信息。			
示例	jMVBCTI. SetDebug (1);			
参数	**名称**	**类型**	**用途**	**参数实例**
	level	数值	调试级别（0-3）	1
返回值	无			

8.51 SetAgent 设置坐席工号与设备号

函数原型	SetAgent (parm)			
功能	设置坐席，不支持多坐席，仅可调用一次。			
示例	jMVBCTI. SetAgent ({agentno:'7001'}); jMVBCTI. SetAgent ({agentno:'7001',deviceno:'7001'});			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
parm 对象成员	agentno	string	工号，必须	7001
	deviceno	string	设备号，可选，默认与工号相同	6001
	callerid	string	呼出显示号码，可选	
返回值	无			

8.52 SetQueue 设置队列号

函数原型	SetQueue (parm)			
功能	设置队列，支持多队列,多次调用增加多个队列。			
示例	jMVBCTI. SetQueue ({queueno:'7970'}); jMVBCTI. SetQueue ({queueno:'7970',penalty:'0'});			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
parm 对象成员	queueno	string	队列号，必须	7970
	penalty	string	坐席优先级，可选，默认 0	0
返回值	无			

8.53 SetNotify 设置消息提示开关

原型	SetNotify(b)			
功能	设置消息提示开关，需在 Start 前调用。控制接口是否以 alert 方式输出警告信息。			
示例	jMVBCTI.SetNotify(true);			
参数	名称	类型	用途	参数实例
	B	字符串	布尔值(true/false)	true
返回值	无			

8.54 SetVars 批量设置通道变量

函数原型	SetVars (parm, callback)			
功能	批量设置通道变量。			
示例	jMVBCTI. SetVars ({channel: ' SIP/100-102-0a005df0', variable0: 'MVB_CALLEE, value0:' 8001' } ,callback);			
参数	名称	类型	用途	参数实例
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，必须	
parm 对象成员	channel	string	活动通道 ， 必须	SIP/100-102-0a005df0
	variable0- variable9	string	变量名 0-9，至少一项	MVB_CALLEE
	value0- value9	string	变量值 0-9，至少一项	8001
回调函数原型	callback (resdata)			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息，需要 decodeURI	resdata. message
	data	json 对象	存放变量赋值结果	resdata.data.varname
data 对象集合	variable0-9	string	变量名 0-9 的赋值结果，成功：Success	resdata.data.variable0 resdata.data.variable9 Success

8.55 SetWebKey 设置认证密钥

原型	SetWebKey (webkey)			
功能	设置认证密钥，需在 Init 后，使用接口函数前调用。webkey 的获取方法详见“认证密钥”章节。			
示例	jMVBCTI. SetWebKey (webkey);			
参数	名称	类型	用途	参数实例
	webkey	string	认证密钥	e10adc3949ba59abbe56e057f20f883e

返回值	无			

8.56 SendSMS 发短信

函数原型	SendSMS (parm, callback)			
功能	拨打指定电话，采用先呼坐席模式，坐席摘机后才会拨打指定号码。			
示例	jMVBCTI. SendSMS ({phoneno :'189063979XX', smstext:'您好,畅信达' }); jMVBCTI. SendSMS ({phoneno :' 189063979XX', smstext:' 您好,畅信达' } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，可选	
parm 对象成员	phoneno	string	号码，必须	189063979XX
	smstext	string	短信内容，必须	您好,畅信达
	reqdate	string	预约日期，可选	2012-07-01
	starttime	string	预约开始时间 可选	19:30:10
	endtime	string	预约结束时间 可选	22:30:10
	appid	string	应用 id 可选 最大 32 字符	CSR2000
	appflag	string	应用处理标记 可选 最大 32 字符	new
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.57 Split 分离工号与设备

函数原型	Split (parm, callback)			
功能	分离工号与设备。			
示例	jMVBCTI. Split ( ); jMVBCTI. Split ({},callback); jMVBCTI. Split ({agentno: '7001'} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
parm 对象成员	callback	function	回调函数，可选	
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode

	message	string	错误信息,需要 decodeURI	resdata. message

8.58 Spy 监听/强插/密语

函数原型	Spy (parm, callback)			
功能	对指定号码或活动通道实现监听/强插/密语。 监听:监听者可以听到通话双方的内容，通话双方听不到监听者声音。 密语：监听者可与被监听对象交流，与被监听对象通话的一方听不到监听者声音。 强插:监听者可以插入通话，类似 3 方通话效果。 接通后默认模式为监听，监听者可以在话机上按键进行切换： 4-监听 5-密语 6-强插			
示例	jMVBCTI. Spy ({activechannel : ' SIP/100-102-0a005df0' }); jMVBCTI. Spy ({activechannel : ' SIP/100-102-0a005df0' } ,callback); jMVBCTI. Spy ({spyexten: ' 7002' }); jMVBCTI. Spy ({spyexten: ' 7002' } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	activechannel	string	活动通道 2 选 1	SIP/100-102-0a005df0
	spyexten	string	号码 2 选 1	7002
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata. errcode
	message	string	错误信息,需要 decodeURI	resdata. message

8.59 TransferToPhone 转接到指定号码

函数原型	TransferToPhone (parm, callback)			
功能	转接到指定的电话号码，默认采用询问转接，坐席方与第三方先通话，客户听音乐，坐席挂机后客户与第三方接通。若第三方先挂机，则坐席恢复与客户通话。			
示例	jMVBCTI. TransferToPhone ({phoneno :' 189063979XX' }); jMVBCTI. TransferToPhone ({phoneno :' 189063979XX' } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，必须	
	callback	function	回调函数，可选	
parm 对象成员	phoneno	string	号码，必须	189063979XX
	blind	string	盲转标志(0/1),可选 默认:0	您好,畅信达
	callindex	number	呼叫索引 可选。用以区分坐席进行的多路活动通话。 默认:0	0

	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
	endtime	string	预约结束时间 可选	22:30:10
	appid	string	应用 id 可选 最大 32 字符	CSR2000
	appflag	string	应用处理标记 可选 最大 32 字符	new
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.60 TransferToIVR 转接到指定 IVR

函数原型	TransferToIVR (parm, callback)			
功能	转接到指定语音导航，默认转接到主菜单。			
示例	jMVBCTI. TransferToIVR (); jMVBCTI. TransferToIVR ({context:’ivr-2, exten : ’ s’ , priority: ’1’ } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	context	string	入口，可选 默认：from-trunk	ivr-1
	exten	string	号码,可选 默认:s	s
	priority	string	序号,可选 默认:1	1
	callindex	number	呼叫索引 可选。用以区分坐席进行的多路活动通话。 默认:0	0
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.61 TransferToFax 转接到电子传真中心

函数原型	TransferToFax (parm, callback)			
功能	转接到电子传真中心。			
示例	jMVBCTI. TransferToFax (); jMVBCTI. TransferToFax ({} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	callindex	number	呼叫索引 可选。用以区分坐席进行的多路活动通话。 默认:0	0
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.62 UnHold 坐席取消闭音

函数原型	UnHold (parm,callback)			
功能	坐席取消闭音，分机与对方通话恢复			
示例	jMVBCTI. UnHold ( ); jMVBCTI. UnHold ({},callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	callback	function	回调函数，可选	
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.63 UnPause 坐席示闲

函数原型	UnPause (parm, callback)			
功能	坐席示闲。			
示例	jMVBCTI. UnPause ( ); jMVBCTI. UnPause ({},callback); jMVBCTI. UnPause ({queueno: ’ 7910’} ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**

	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	queueno	string	队列号码 ， 可选	7910
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

8.64 UnPark 取消呼叫保持（驻留）

函数原型	UnPark (parm, callback)			
功能	取消保持通话。			
示例	jMVBCTI. UnPark (); jMVBCTI. UnPark ({exten: ’ 701’ } ,callback); jMVBCTI. UnPark ({agentno: ‘7001’ , exten: ’ 702’ } ,callback);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	exten	string	位置号，可选	701
	callerid	string	回叫坐席时显示的号码。 默认:原主叫号码	空
	agentno	string	坐席号码,可选 默认为调用 SetAgent 加入的坐席	7001
回调函数原型	callback (resdata) 也可以不指定回调函数			
回调函数参数	名称	类型	用途	参照方法
	errcode	number	错误代码，0 代表成功	resdata.errcode
	message	string	错误信息,需要 decodeURI	resdata.message

9 事件

9.1 OnHeart 轮询事件

事件绑定	OnHeart(callback)			
功能	设置轮询事件回调函数。			
示例	jMVBCTI.OnHeart(cti_onheart_handler)			
回调函数原型	cti_ onheart _handler()			
回调函数参数	名称	类型	用途	参数实例
	无			
返回值				

9.2 OnAjaxError 设置 http 请求错误处理事件

事件绑定	OnAjaxError(callback)			
功能	设置 http 请求错误事件回调函数。			
示例	jMVBCTI. OnAjaxError(cti_onajaxerror_handler)			
回调函数原型	cti_onajaxerror_handler(XMLHttpRequest, textStatus, errorThrown)			
回调函数参数	名称	类型	用途	参数实例
	XMLHttpRequest	object	xhr 对象	
	textStatus	text	错误信息,除了得到 null 之外,还可能是"timeout","error","notmodified"和"parsererror"。	
	errorThrown	object	捕获的异常对象,通常 textStatus 和 errorThrown 之中只有一个会包含信息	
返回值				

9.3 OnStarted 启动完成事件

事件绑定	OnStarted (callback)			
功能	设置启动完成事件回调函数。			
示例	jMVBCTI. OnStarted (cti_onstarted_handler)			
回调函数原型	cti_ onstarted _handler()			
回调函数参数	名称	类型	用途	参数实例
	无			
返回值				

9.4 OnCalled 弹屏事件

事件绑定	OnCalled (callback)
功能	设置启动完成事件回调函数。

示例	jMVBCTI. OnCalled (cti_oncalled_handler)			
回调函数原型	cti_oncalled_handler(user,device, channel, popupcalls)			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	user	string	坐席工号	7001
	device	string	设备号	6001
	channel	string	设备接口	SIP/6001
	popupcalls	object	弹屏数据对象	详见数据对象部分 此处的 popupcall 不是对象集合，使用 popupcall.varname 方式访问数据，不是 popupcalls[i].varname
返回值	无			

9.5 OnConference 会议状态变化事件

事件绑定	OnConference (callback)			
功能	设置会议状态变化事件回调函数。			
示例	jMVBCTI. OnConference (cti_ onconference _handler)			
回调函数原型	cti_ onconference _handler (user,device, channel, confnos)			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	user	string	坐席工号	7001
	device	string	设备号	6001
	channel	string	设备接口	SIP/6001
	popupcalls	object	会议数据对象	详见数据对象部分 confnos[0].confno confnos[0].usernum
返回值				

9.6 OnStatus 设置设备状态变化事件

事件绑定	OnStatus (callback)			
功能	设置设备状态变化事件回调函数。			
示例	jMVBCTI. OnStatus (cti_ onstatus _handler)			
回调函数原型	cti_oncalled_handler(user,device, channel, devicestatus)			
回调函数参数	**名称**	**类型**	**用途**	**参数实例**
	user	string	坐席工号	7001
	device	string	设备号	6001
	channel	string	设备接口	SIP/6001
	devicestatus	number	设备状态	详见常数说明 使用 GetStatusDesc 可以获取中文描述
返回值				

9.7 OnQueuePause 设置坐席示忙示闲事件

事件绑定	OnQueuePause (callback)			
功能	设置坐席示忙/示闲事件回调函数。			
示例	jMVBCTI. OnQueuePause (cti_ onqueuepause _handler)			
回调函数原型	cti_onqueuepause_handler (agentno,device,channel,queueno,ismember,paused)			
回调函数参数	名称	类型	用途	参数实例
	user	string	坐席工号	7001
	device	string	设备号	6001
	channel	string	设备接口	SIP/6001
	queue	string	队列号	90
	ismember	string	成员标志(0/1)	1
	paused	string	示忙标志(0/1)	1
返回值				

9.8 OnQueueMember 设置坐席上下线事件

事件绑定	OnQueueMember (callback)			
功能	设置坐席上线、下线事件回调函数。			
示例	jMVBCTI. OnQueueMember (cti_onqueuemember_handler)			
回调函数原型	cti_onqueuepause_handler(agentno,device,channel,queueno,ismember,paused)			
回调函数参数	名称	类型	用途	参数实例
	user	string	坐席工号	7001
	device	string	设备号	6001
	channel	string	设备接口	SIP/6001
	queue	string	队列号	90
	ismember	string	成员标志(0/1)	1
	paused	string	示忙标志(0/1)	1
返回值				

9.9 OnGUIRefresh 设置界面更新事件

事件绑定	OnGUIRefresh (callback)			
功能	设置界面更新事件回调函数。			
示例	jMVBCTI. OnGUIRefresh (cti_onguirefresh_handler)			
回调函数原型	cti_ onguirefresh _handler(Agents)			
回调函数参数	名称	类型	用途	参数实例
	Agents	object	坐席对象集合	详见数据对象部分
返回值				

10 jMVBCTI_Bar 便捷工具条

10.1 启用工具条方法

函数原型	Bar(x,y,divstatus,url,width,height)			
功能	开启便捷工具条			
示例	jMVBCTI_Bar.Bar(30,30,0,'http://www.ipxchina.cn/index.html',500,200);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	x	int	工具条距左边距离	30
	y	int	工具条距上边距离	30
	divstatus	int	工具条样式,0/1(横版/竖版)	0
	url	string	弹屏窗口的 url	http://www.ipxchina.cn/index.html
		url 被访问时，自带以下四个参数		
		callerid	主叫号码	http://www.ipxchina.cn/index.html?callerid=4000820188&callee=601&uniqueid="1590649180.76&refuniqueid=1590649180.77
		callee	坐席号码	
		uniqueid	坐席通道呼叫编号	
		refuniqueid	对方呼叫编号	
	width	int	弹屏窗口的宽度	500
	height	int	弹屏窗口的高度	200

10.2 加载外部 CSS 样式文件方法

函数原型	setCss(css)			
功能	加载外部样式文件，可以自定义工具条内按钮样式			
示例	jMVBCTI_Bar.setCss(“bar_css.css”);			
参数	**名称**	**类型**	**用途**	**参数实例**
	parm	json 对象	传递参数，可选	
	callback	function	回调函数，可选	
parm 对象成员	css	string	Css 样式文件名	bar.css

11 数据对象

11.1 Agents 坐席对象集合

Agents 对象集合是 jMVBCTI 接口在浏览器内存中维护的一个本地对象集合，由 Agent 对象构成的数组，可通过 Agents[0]。当前版本仅支持唯一对象，不再支持多座席，可以使用 Agents[0]访问其内容。
原型：Agent[]

11.2 Agent 坐席对象

多个 Agent 对象构成 Agents 对象集合。Agent 对象包含坐席、状态、队列、呼叫、弹屏等数据或对象。

成员	类型	用途	参数实例
user	string	用户号(工号)	Agents[0].user 7001
device	string	设备号	Agents[0]. device 6001
channel	string	拨号接口	SIP/6001
devicestatus	string	设备状态 详见<附表.坐席线路状态>	0
devicestatuslast	number	上次设备状态 详见<附表.坐席线路状态>	4
devicetype	number	设备绑定类型 详见<附表.设备绑定类型>	fixed
devicecontext	string	设备拨号权限	from-internal-1
callerid	string	主叫号码	8001
isbind	bool	是否已绑定(true/false)	false
confnos[]	object	会议状态	详见对象说明
confnoslast[]	object	上次会议状态	
popupcalls[]	object	弹屏数据对象	
calls[]	object	通话数据对象	
queues[]	object	队列数据对象	
queueslast[]	object	上次队列数据对象	
parks[]	object	驻留数据对象	

11.3 confnos 会议数据对象

存放坐席会议数据，内含坐席通过 CTI 发起的、通过电话进入的会议室列表。

- 坐席退出会议室后，如果会议未关闭，此对象中任然返回该会议室号码，但与会者编号

为空。

- 通常用户只需要响应会议状态变化事件即可，不需要访问此对象。
- 由多条记录构成，每条结构如下:

成员	类型	用途	参数实例
confno	string	会议室号码	confnos [0]. confno: 998845
usernum	string	坐席在会议室中与会者编号，坐席退出后,此编号为空字符串。	confnos [0]. usernum:2

11.4 confnoslast 上次会议数据对象

存放与坐席相关的上次会议状态数据，多条记录构成，结构同 confnos。

11.5 popupcalls 弹屏数据对象

存放坐席弹屏数据,通常用户只需要响应弹屏事件即可,不需要访问此对象。多条记录构成，每条结构如下:

成员	类型	用途	参数实例
channel	string	坐席活动通道	popupcalls [0]. channel SIP/100-102-08bb8a98
uniqueid	string	坐席通道呼叫编号	popupcalls [0]. uniqueid 1341286895.2
callerid	string	主叫号码	popupcalls [0]. callerid 8001
calleridname	string	主叫名称，需 decodeURI	<Unknown>
refchannel	string	对方活动通道	SIP/7001-08bd95f8
refuniqueid	string	对方呼叫编号	1341286896.3
direct	string	方向: IN 主叫 OUT 被叫	IN
state	string	状态: Ringing 回铃 Ring 振铃 Up 接通	Ringing

queuenum	string	队列号码，客户呼叫进入的队列号码。非来自队列的呼叫，此参数为空字符串。	7910
did	string	原被叫号码，指示客户拨打哪一个外线号码进入呼叫中心，通常用来区分不同的业务及服务。	82754888
ivrhist	string	IVR 导航历史，记录客户呼叫接通坐席前浏览过哪些语音导航。格式：语音导航 id 和减号分隔构成。	18-15-9
holdtime	string	等待时间（秒），指示客户呼叫接通坐席前已在队列排队的时间。	240
isdefault	string	是否默认,用于标示一个坐席发起多路呼叫时，最后接通的为择默认。	1

11.6 calls 通话数据对象

存放坐席当前通话数据，多条记录构成，每条结构如下：

成员	类型	用途	参数实例
createtime	string	数据建立时间	calls[0].createtime 2012-07-02 14:08:39
channel	string	坐席活动通道	calls[0]. channel SIP/7001-08bd95f8
refchannel	string	对方活动通道	calls[0].refchannel SIP/100-102-08bb8a98
callstate	string	状态 Dialing 拨号 Link 接通	calls[0].callstate Link
callerid	string	号码	calls[0]. callerid 7001
calleridname	string	名称	
refcallerid	string	对方号码	8001
refcalleridname	string	对方名称	
channelstate	string	坐席活动通道状态	Up

refchannelstate	string	对方活动通道状态	Up
uniqueid	string	坐席通道编号	1341286896.3
refuniqueid	string	对方通道编号	1341286896.5
dialtime	string	拨号时间	2012-07-02 14:08:39
linktime	string	接通时间	2012-07-02 14:30:39
direct	string	呼叫方向 IN 被叫 OUT 主叫	OUT

11.7 queues 队列数据对象

存放与坐席相关队列数据，多条记录构成，每条结构如下：

成员	类型	用途	参数实例
queueno	string	队列号	queues[0].queueno 7910
ismember	string	坐席是否上线 0-否 1-是	queues[0]. ismember 1
agentname	string	坐席工号	queues[0]. agentname 7001
agent	string	坐席接口	queues[0]. agent SIP/6001
agentship	string	坐席类型 static 固定 dynamic 动态	queues[0]. agentship
penalty	string	坐席优先级	queues[0]. penalty 0
paused	string	是否示忙 0-否 1-是	queues[0]. paused 1

11.8 queueslast 上次队列数据对象

存放与坐席相关的上次队列数据，多条记录构成，结构同 queues。

11.9 parks 呼叫保持数据对象

存放与坐席相关呼叫保持数据，多条记录构成，每条结构如下：

成员	类型	用途	参数实例
createtime	string	数据建立时间	parks[0]. createtime 2012-07-02 14:08:39
exten	string	驻留位置	parks [0]. exten 701
channel	string	驻留活动通道	parks [0]. channel SIP/100-102-08bb8a98
srcchannel	string	坐席活动通道	parks [0]. srcchannel SIP/7001-08bd95f8
timeout	string	最大驻留时间	parks [0]. timeout 60
callerid	string	来电号码	parks [0]. callerid 8001
calleridname	string	来电名称	parks [0]. calleridname
state	string	状态	parks [0]. state Parked
parkedtime	string	驻留时间	parks [0]. parkedtime 2012-07-02 14:08:35

12 附表

12.1 设备绑定类型

值	描述
fixed	固定关系(不允许分离)
adhoc	临时关系(允许分离)

12.2 坐席类型

值	描述
static	固定
dynamic	动态

12.3 坐席线路状态

值	描述
0	空闲
1	使用
2	正忙
4	离线
8	振铃
9	振铃
-1	故障

12.4 发起呼叫返回 Reason 说明（同步方式有效）

值	描述
1	被叫挂机
4	接通
5	被叫忙
8	线路忙

13 实例

13.1 引入 jMVBCTI 库

```
<script                                                    type="text/javascript"
src="http://192.168.0.218/webservice/jmvbcti/jquery.js" ></script>
<script                                                    type="text/javascript"
src="http://192.168.0.218/webservice/jmvbcti/jMVBCTI.js" ></script>
```

13.2 获取认证密钥

```
/*计算密钥
此示例仅作演示用途，考虑到安全性，认证密钥的计算最好在后台完成。
*/
function cti_createwebkey(keystring)
{
  var today=new Date();
    date=today.getDate();
    if(date<=9) date="0"+date;
    month=today.getMonth();
    month=month+1;
    if(month<=9) month="0"+month;
    year=today.getFullYear();
    var nowDate=year+'-'+month+'-'+date;
    var webstring = MD5(keystring+nowDate);
    return webstring;
}
var mvb2000webkey= cti_createwebkey (webkey);
```

13.3 初始化

```
14  忽略事件
jMVBCTI.Init("http://192.168.0.218");
jMVBCTI.SetWebKey("e5ad7e38d68ebe4628d20ce3351ac6bc  "); //根据认证密钥规范计算
获得
jMVBCTI. SetAgent ({agentno:'7001',deviceno:'7001'});
jMVBCTI. SetQueue ({queueno:'7970',penalty:'0'});
jMVBCTI.Start();

15  响应事件
jMVBCTI.Init("http://192.168.0.218");
jMVBCTI.SetWebKey("e5ad7e38d68ebe4628d20ce3351ac6bc  "); //根据认证密钥规范计算
```

获得

```
jMVBCTI. SetAgent ({agentno:'7001',deviceno:'7001'});
jMVBCTI. SetQueue ({queueno:'7970',penalty:'0'});
jMVBCTI.OnHeart(cti_onheart_handler);
jMVBCTI.OnStarted(cti_onstarted_handler);
jMVBCTI.OnCalled(cti_oncalled_handler);
jMVBCTI.OnStatus(cti_onstatus_handler);
jMVBCTI.OnQueueMember(cti_onqueuemember_handler);
jMVBCTI.OnQueuePause(cti_onqueuepause_handler);
jMVBCTI.OnGUIRefresh(cti_onguirefresh_handler);
jMVBCTI.Start();
```

15.1 拨号

1、无回调

```
jMVBCTI.Dial( '4000820188' );
```

2、有回调

```
jMVBCTI.Dial( '4000820188' ,function(resdata){
    if(resdata.errcode==0)alert( '成功' );
    else )alert(decodeURI(resdata.message));
});
```

15.2 来电弹屏事件

```
jMVBCTI.OnCalled(cti_oncalled_handler);
//来电事件
function cti_oncalled_handler(agentno,device,channel,popupcall){
      var obj=$('#calleriddiv > span');
      var calleridinfo=popupcall.direct+' '+popupcall.callerid;
        obj.text(calleridinfo);
}
```

15.3 设备状态变化事件

```
//设备状态变化事件
function cti_onstatus_handler(agentno,device,channel,status){
     var obj=$('#devicediv > span');
     var                    statusinfo=agentno+'                    '+channel+'
'+jMVBCTI.GetStatusDesc(status)+'('+status+')';      obj.text(statusinfo);
}
```

15.4 绑定号码与设备

```
jMVBCTI.Bind();
```

15.5 分离号码与设备

```
jMVBCTI.Split();
```

15.6 坐席上线

```
jMVBCTI.OnLine();
jMVBCTI.OnLine({queueno: '7910' });
```

15.7 坐席下线

```
jMVBCTI.OffLine();
jMVBCTI. OffLine ({queueno: '7910' });
```

15.8 坐席示忙

```
jMVBCTI.Pause();
jMVBCTI.Pause({pausetype:' 2' });
jMVBCTI.Pause({queueno:' 7910' ,pausetype:' 2' });
```

15.9 坐席示闲

```
jMVBCTI.UnPause();
```

15.10 拆线

```
jMVBCTI.HangUp();
```

15.11 批量设置变量

```
jMVBCTI.SetVars({channel: ' SIP/7001-
082c9f38 ' ,variable0:'MVB_Var1',value0:'4000820188',variable1:'MVB_Var2',
value1:'ipxchina'} ,function(resdata){
alert(resdata.errcode);
});
```

15.12 批量获取变量

```
jMVBCTI.GetVars({channel:                    '                                    SIP/7001-
082c9f38' ,variable0:'MVB_Var1',variable1:'MVB_Var2'} ,function(jsondata){
    alert('MVB_Var1:'+jsondata.data.MVB_Var1+'
MVB_Var2:'+jsondata.data.MVB_Var2);
});
```

15.13 建立 3 方通话

```
function cti_confcreate(){
   var curcalls=jMVBCTI.GetCalls();
   if (typeof curcalls == 'undefined' || curcalls.length == 0)
    {
    alert('坐席尚未建立通话!');
    return ;
    }
    getPhonenoWindow('getinput','请输入电话号码',function(phonenolist){
      var activechannel1=curcalls[0].channel;
      var activechannel2=curcalls[0].refchannel;
      var
parm={conftype:'D',phonenolist:phonenolist,async:'0',activechannel1:activechann
el1,activechannel2:activechannel2};
      jMVBCTI.ConfCreate(parm,function(resdata){
    try{
       if(resdata.errcode==0)
        alert('三方请求已发送,会议号:'+resdata.data.confno);
       else
         alert('发起三方请求失败:'+decodeURI(resdata.message));
       }catch(e){}
    });
    });
    }
```

15.14 退出或终止会议

```
    function cti_confexit    (){
        jMVBCTI.ConfExitAll({exitall:'0'},function(jsondata){
            alert("已退出会议"+jsondata.data.confno);
        });
    }
    function cti_confoff(){
        jMVBCTI.ConfExitAll({exitall:'1'},function(jsondata){
            alert("已终止会议"+jsondata.data.confno);
        });
    }
```

15.15 获取录音播放 URL

```
   jMVBCTI.GetMonitorURL({uniqueid:uniqueid},function(resdata){
    try{
        if(resdata.errcode==0){
            if(resdata.data.length>0){
    $('#monitorurl').val(jMVBCTI.GetMVBServerURL()+resdata.data[0]);
                playfile(jMVBCTI.GetMVBServerURL()+resdata.data[0]);
            }else
                alert('无录音文件');
        }else
            alert('获取录音播放 URL 列表失败:'+decodeURI(resdata.message));
    }catch(e){}
});
```

15.16 调用 WEBCTI 高级接口

```
//获取号码归属
function cti_getarea(phoneno){
    var ctiaction='getarea.php';
    var parm={phoneno:phoneno};
    jMVBCTI.CallWebCTI(ctiaction,parm,function(jsondata){
    try{
        var rows= jsondata.root.row;
        if(typeof rows!='undefined' && rows.length>0)
            alert(rows[0].area+'  '+rows[0].province+  '  '+rows[0].city+  '
'+rows[0].type);
        }catch(e){}
    });
}
```