MVB2000 WEBCTI2 接口使用方法

部门 ： 产品部
版本/创建时间 ：2.0.0.3/2021-09-09

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：MVB2000 WEBCTI2 接口使用方法

存放位置： SVN

编辑记录：

版本	日期	编写者	编辑原因
V2.0.0.0	2015-2-4	畅信达	创建此文档
V2.0.0.1	2015-3-19	畅信达	修改录音获取接口,增加mp3 支持
V2.0.0.2	2019-11-01	畅信达	修改排版，添加部分文档说明
V2.0.0.3	2021-09-09	畅信达	公司信息变更

目录

1 快速入门

WEBCTI 接口功能丰富，几乎涵盖所有 MVB2000 开发接口。但在实际应用中，不同的需求需要实现的功能亦不同，使用到的接口也不同。每个用户可能只会使用其中的一部分接口。下面以最常见的应用场景举例说明，逐步引导开发者熟悉 WEBCTI 接口，开发出满足自己需求的呼叫中心应用。

1.1 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。

1.2 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 WEBCTI 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。

1.3 设置与测试接口

1.3.1 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。

1.3.2 计算密钥参数

根据《MVB2000_WEB 开发接口》的 webkey 设置部分计算密钥参数。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）。此处假设计算好的密钥参数为：5589a16d0b48a49ce1c66c2ad11268c0。

1.3.3 MD5 计算参考值

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

1.3.4 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）

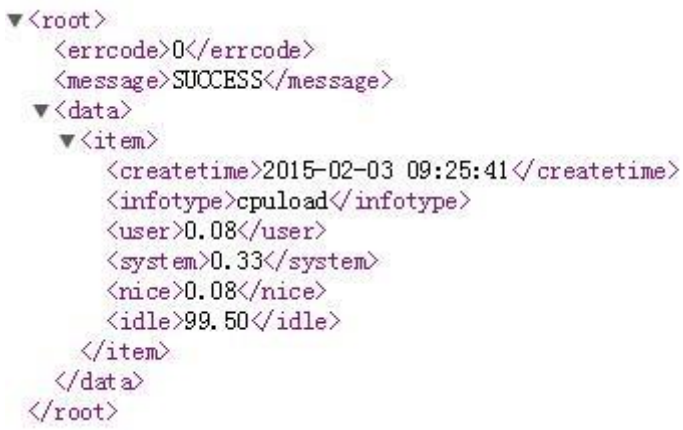
三、打开浏览器，在地址栏输入：

http://192.168.0.201/webservice2/getcpuload.php?webkey=5b3c8c39f0ec17e1823e973c23fad009

如果您使用 javascript 编程，习惯使用 json，也可以输入:
http://192.168.0.201/webservice2/getcpuload.php?webkey=5b3c8c39f0ec17e1823e97

3c23fad009&json=2

若返回信息：



说明接口调用成功。

若返回信息：{"errcode":9999,"message":"WebKey 5b3c8c39f0ec17e1823e973c2fad009 is invalid","data":[]}

说明密钥参数错误。

1.4 调用接口方法

在 windows 应用程序中调用

对于开发 CS 架构呼叫中心应用或 CRM 系统的程序员，您有 3 种选择：

- 通过 HTTPRequest 使用 Get 方式调用 WEBCTI 接口，然后使用 XMLDocument 直接解析返回的 XML 对象。
- 使用我公司提供的 MVBCTI ActiveX 控件普及版，使用 OCX 控件提供的方法和事件与 MVB2000 平台交互。但普及版只提供少量功能，如果您只需要来电弹屏和点击拨号功能，可以使用普及版。普及版免费提供，不收取费用。
- 使用我公司提供的 MVBCTI ActiveX 控件专业版，使用 OCX 控件提供的方法和事件与 MVB2000 平台交互。使用该控件可实现 MVB2000CTI 控制、传真收发、短信收发、mysql 数据库访问等功能。MVBCTI ActiveX 控件专业版是一套独立的产品，您需要单独购买。

以上 3 种方式都能满足应用开发需求，选择哪种方式取决与您的编程喜好和业务需求。

本文档讨论第一种调用方式(WEBCTI)：

1.4.1 使用 VB6 调用 WEBCTI（后台调用)

```
Dim xmlHttp As MSXML2.xmlHttp
Dim xmlobject As IXMLDOMNode
Dim URL As String
Dim res As String
Dim rows As Integer
Set xmlHttp = New MSXML2.xmlHttp
webkey = get_webkey("0866089e569bbd3d") '需要按照要求实现此函数
URL = "http://192.168.0.102/webservice/getcpuload.php?webkey=" + webkey + "&uniqueid=1294228771.15"

xmlHttp.open "GET", URL, False
xmlHttp.send

Set xmlobject = xmlHttp.responseXML
Dim nodelist As IXMLDOMNodeList

Set nodelist = xmlobject.selectNodes("//row")

If nodelist.length = 0 Then
    MsgBox "无数据"
End If
For i = 0 To nodelist.length - 1
    res = res + "createtime=" + nodelist.Item(i).selectSingleNode("createtime").nodeTypedValue + Chr(13) + Chr(10)
    res = res + "infotype=" + nodelist.Item(i).selectSingleNode("infotype").nodeTypedValue + Chr(13) + Chr(10)
    res = res + "user=" + nodelist.Item(i).selectSingleNode("user").nodeTypedValue + Chr(13) + Chr(10)
    res = res + "system=" + nodelist.Item(i).selectSingleNode("system").nodeTypedValue + Chr(13) + Chr(10)
    res = res + "nice=" + nodelist.Item(i).selectSingleNode("nice").nodeTypedValue + Chr(13) + Chr(10)
    res = res + "idle=" + nodelist.Item(i).selectSingleNode("idle").nodeTypedValue + Chr(13) + Chr(10)
Next

Text1.Text = res
```

使用此方法可建立 windows 应用程序。

1.4.2 使用 C# 调用 WEBCTI（后台调用)

```
/*通过 web 接口获取返回 xml 数据*/
 private XmlDocument get_ctixml(string url) {
            XmlDocument doc = new XmlDocument();
             HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);
             request.UnsafeAuthenticatedConnectionSharing = true;
             request.Method = "GET";
             request.Timeout = 5000;
             // Downloads the XML file from the specified server.
             try
             {
                  HttpWebResponse response = (HttpWebResponse)request.GetResponse();
                  StreamReader sr = new StreamReader(response.GetResponseStream(), System.Text.Encoding.GetEncoding("gb2312"));
                 String retXml = sr.ReadToEnd();
                 sr.Close();
                 doc.LoadXml(retXml);
                 response.Close();
             }
             catch (Exception exc)
             {
                  System.Diagnostics.Debug.WriteLine(exc.StackTrace);
                  MessageBox.Show(exc.Message);
             }
             return doc;
     }
     /*解析 xml 数据*/
     private string get_xmlvalue( XmlNode Node,string feild) {

          XmlNode xn = Node.SelectSingleNode(feild);
          if (xn == null || xn.FirstChild ==null)
               return "";
          else
          {
               return xn.FirstChild.Value;
          }
     }
```

```
   /*计算 WEBKEY 参数*/
  private string get_webkey()
     {

          string strResult="";
          string mvbwebkey = "";
          DateTime currentTime = DateTime.Now;
          mvbwebkey = tb_webkey.Text;
          if (mvbwebkey.Length == 0)
          {
               MessageBox.Show("请输入 MVB2000 平台的 WEB 密钥!");
               tb_webkey.Focus();
          }
           MD5 md5=MD5.Create();
           mvbwebkey = mvbwebkey + currentTime.ToString("yyyy-MM-dd");

           byte[] bytResult = md5.ComputeHash(System.Text.Encoding.ASCII.GetBytes(mvbwebkey));
          for (int i = 0; i <bytResult.Length; i++){

               string hexOutput = String.Format("{0:x2}", bytResult[i]);

               strResult = strResult + hexOutput;
       }
            md5.Clear();
            tb_key.Text = strResult;
     return strResult;
     }


     /*获取设备拨号接口*/
     private string   get_interface(string strdevice)
     {
          string url;

          string webkey = get_webkey();
          if (webkey.Length == 0) return null;
          url = tb_web.Text + "/webservice/ctigetdevice.php?webkey=" + webkey + "&device=" + strdevice;
          XmlDocument xmldoc = get_ctixml(url);
          XmlNodeList nodelist = xmldoc.GetElementsByTagName("row");
          if (nodelist == null || nodelist.Count == 0) return null;
          XmlNode row = nodelist.Item(0);
          int feildsnum = row.ChildNodes.Count;
          if (feildsnum < 2) return null;
          if (Convert.ToInt32(get_xmlvalue(row, "errcode")) == 0)
          {
              return   get_xmlvalue(row, "dial");
          }
          return null;

     }
```

使用此方法可建立 windows 应用程序、或者作为 WEB 后台程序使用。

1.4.3 使用 PHP 调用 WEBCTI（后台调用)

```
<?php
/*使用 PHP 后台调用 MVB2000 WEBCTI 接口*/
Class MVB2000_WEBCTI {
        var $xml_result =array();
        var $xml_elem = null;

        function xml_startElement( $parser, $name, $attrs )
        {
          global $g_rows, $g_elem;
          if ($name == 'ROW'){
                $this->xml_result[]=array();
          }
          $this->xml_elem = $name;
        }
        function xml_endElement( $parser, $name )
        {
            $this->xml_elem = null;
        }
        function xml_textData( $parser, $text )
        {

          $this->xml_elem=trim($this->xml_elem);
          if ( $this->xml_elem != 'ROOT' && $this->xml_elem != 'ROW' && strlen($this->xml_elem)>0 ){
                $i =count($this->xml_result);
                if( $i > 0 )$i=$i-1;
                $this->xml_result[$i][$this->xml_elem] = $text;
          }
        }
        function xml_parser($url){
                $this->$xml_result=array();
                $this->$xml_elem = null;
                $parser = xml_parser_create();
                xml_parser_set_option($xml_parser, XML_OPTION_CASE_FOLDING, true);
                xml_set_element_handler($parser, array(&$this, 'xml_startElement'),array(&$this, 'xml_endElement'));
                xml_set_character_data_handler($parser, array(&$this, 'xml_textData'));
                $f = fopen( $url, 'rb' );
                if ($f===false){
                                $this->xml_result[0]['xmlerrcode']='1';
                                $this->xml_result[0]['xmlerrmsg']='访问 WEBCTI 接口失败';
                        return $this->xml_result;
                }
                while( $data = fread($f,4096))
                {
                        if(xml_parse( $parser, $data )===false){
                                $errcode=xml_get_error_code ($parser);
                                $errmsg=xml_error_string ($errcode);
                                $this->xml_result[0]['xmlerrcode']='2';
                                $this->xml_result[0]['xmlerrmsg']="($errcode)".$errmsg;
                                return $this->xml_result;
                        }
                }
                xml_parser_free( $parser );
                return $this->xml_result;
        }
        function xml_webkey($webkey)
        {
            return md5($webkey.date('Y-m-d'));
        }
}

$WEBCTI=new MVB2000_WEBCTI();
$webkey='0866089e569bbd3d';
$webhost='http://192.168.0.218';
$apiname="getqueuemember.php";
$queue='7910';

$newkey=$WEBCTI->xml_webkey($webkey);
$url="{$webhost}/webservice/{$apiname}?webkey={$newkey}&queue={$queue}";
$res=$WEBCTI->xml_parser($url);
echo "返回数据:<br \>";
print_r($res);
?>
```

1.4.4 使用 Javascript 调用 WEBCTI（前台调用）

通过 Ajax 方式调用 web 接口，然后解析返回的 XML 或 JSON 数据，详见 js 示例。
使用 Ajax 方式直接调用接口会碰到跨域权限问题，如果不想采用后台转发的方式解决，建议使用 javascript 标签和 json 返回的方式调用，如：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>通过 SCRIPT 标签调用 webcti 接口</title>
<script language="Javascript">
var mvbwebkey="4b5d3985dc5a4e95";     //平台基本设置里显示的 web 密钥
var resjsontype='&json=1';          //返回的 json 数据类型
var httpstring="http://demo.mvb2000.cn:8888/";
/*使用标签 1,获取磁盘信息*/
function webcti_getdiskinfo()
{
      var querystring="webkey="+webcti_createwebkey(mvbwebkey)+resjsontype;
      var urlstring=httpstring+"webservice/getdiskinfo.php?"+querystring
      webcti_do(urlstring,callback_diskinfo);
}
/*在标签 1 加载 url*/
function webcti_do(urlstring,callback)
{
      var t= Math.random();
                  urlstring=urlstring+"&random="+t;
      if(typeof scriptBlock1 == "undefined"){
            scriptBlock1 = document.createElement("script");
      }else{
                  scriptBlock1.src = "";
                  document.getElementsByTagName("head")[0].removeChild(scriptBlock1);
      }
      scriptBlock1.src = urlstring;
      webcti_debug(urlstring);
      scriptBlock1.language = "javascript";
      scriptBlock1.setAttribute('type','text/javascript');
      document.getElementsByTagName("head")[0].appendChild(scriptBlock1);
      scriptBlock1.onreadystatechange = callback;
}
/*回调函数，处理磁盘接口返回数据*/
function callback_diskinfo()
{
            var resRows=0;
            document.getElementById("resdivdisk").innerHTML = "";
      if(scriptBlock1.readyState == "loaded")|| (scriptBlock1.readyState == "complete"))
      {
            if(typeof responseJSON!="undefined"){
                        resRows = responseJSON.root.row.length;
                        webcti_debug("返回"+resRows+"行数据!");
                  document.getElementById("resdivdisk").innerHTML = "";
                  resText="<table border=0 width=100% >";
                  //返回多行数据
                  for(i=0;i<resRows;i++){
                              resText=resText+"<tr align='center' clsss='altbg2'>";
                              resText=resText+("<TD>"+responseJSON.root.row[i].createtime+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].filesystem+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].type+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].size+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].used+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].avail+"</TD>");
                              resText=resText+("<TD>"+responseJSON.root.row[i].mountedon+"</TD>");
                              resText=resText+("</tr>");

                        }
                        document.getElementById("resdivdisk").innerHTML=resText+"</table>";
                        resText="";
                        responseJSON = null;

                  }else{
                        document.getElementById("resdivdisk").innerHTML = "对不起，数据未找到!";
                  }
            }

}
/*计算密钥*/
function webcti_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;
}
MD5 类…,详见 demo。
/*输出调试信息*/
function webcti_debug(debugstr)
{
      var allstr=document.getElementById("debugdiv").innerHTML;
      document.getElementById("debugdiv").innerHTML=allstr+"<BR />"+debugstr;
}
</script>
</head><body ><br /><p><button onclick="webcti_getdiskinfo()">手动获取磁盘信息(使用标签 1)</button></p>
<div id="resdivdisk"    style="border: 1px solid    ; padding:3px;" >磁盘信息调用结果</div>
<br /><div id="debugdiv" style="border: 2px solid blue ; padding:3px;" ><center>Debug Info</center></div>
</body></html>
```

完整 demo 代码详见“webcti_jsondemo.html”。

1.5 开发路线图

无论在 web 前台、后台还是 Windows 应用程序中调用 WEBCTI 接口，其思路和方法是一致

的。本文档仅提供开发入门参考。

- 如果坐席工位固定，每个坐席有固定的电脑和话机，可以忽略动态绑定和分离部分，也不需要将用户和设备设置为分离模式。
- 如果要求坐席永久在线，可以直接将分机号码加入队列。程序中也不需要操作登入和登出，也不允许示忙和示闲。

1.5.1 应用程序登录 ID 与工号

可在应用程序中建立登录 ID 与工号对应关系，通常此关系为一对一。根据每个登录 ID 可以确定其工号。

1.5.2 设备号与电脑 IP 地址

可在应用程序中建立电脑 IP 地址与设备号的对应关系，通常此关系为一对一。根据 IP 地址可以获知距离电脑最近的语音设备号码。

1.5.3 配置设备与用户（工号）关系

参见 1.2，将坐席使用的设备类型设置为临时，以便动态绑定工号。

1.5.4 登录系统时的操作

- 使用 ctigetdevice.php 接口查询设备类型、获取拨号接口、拨号权限、绑定工号。根据绑定工号确定是否先做分离动作。
- 使用 ctibinddevice.php 接口绑定工号与设备。绑定成功后，在设备上拨*65 应能听到新的号码。已经绑定的设备需要先分离再绑定。类型为固定绑定的设备不允许绑定和分离。

1.5.5 登录系统后的操作

- 坐席上线

使用 ctiqueueadd.php 接口将设备加入队列，成为该队列的坐席。加入成功后可接听来自队列的电话。加入队列时使用“拨号接口”，此参数通过第一步已经获得。

- 坐席示忙或示闲

使用 ctiqueuepause.php 接口设置坐席示忙或示闲。示忙成功后队列暂时不给改坐席分配电话。示忙或示闲时使用“拨号接口”，此参数通过第一步已经获得。

- 查询弹屏数据

使用 getpopupscreen.php 接口获取每隔 1500 毫秒（广域网络可设置 2000 毫秒）查询弹屏数据（电话号码、设备活动通道、呼叫编号、呼叫方向），如有数据进行弹屏及客户资料查询操作，通过比对上次弹屏是的呼叫编号和本次获取的呼叫编号可防止重复弹屏。

- 查询设备状态

使用 getextensionstatus.php 接口获取设备状态，显示在界面上。此操作可与弹屏查询动作一起进行。

1.5.6 退出系统前的操作

- 坐席下线

使用 ctiqueueremove.php 接口将设备移出队列。移出成功后不会收到来自队列的电话。移出队列时使用“拨号接口”，此参数通过第一步已经获得。

- 分离工号与设备

使用 ctisplitdevice.php 接口分离工号与设备。分离后设备无法呼入，但可以呼出，呼出时无分机号码。是否需要分离根据实际需求而定。坐席下线后就不会收到来自队列的电话，不分离时，设备可作为普通分机使用。

1.5.7 话务管理操作

- 获取当前设备活动通道和对方活动通道

使用 getchannels.php 两个参数都传入当前设备拨号接口，返回数据中如果状态不是 HangUp，说明通道在使用，channel 以拨号接口开头的说明当前设备为主叫，refchannel 即为对方活动通道。反之，说明当前设备为被叫，channel 即为对方活动通道。
活动通道在转接、拆线等操作均会使用。

- 拆线

需要使用 getchannel.php 接口获取当前设备的活动通道号。或直接使用弹屏数据的活动通道号。

使用 ctihangup.php 接口实现拆线操作。

- 拨号

使用 ctioriginate.php 接口实现外拨操作。详见 3。

- 转接

使用 ctiredirect.php 接口实现转接操作。详见 20。

1.5.8 监控管理操作

- 获取拨号接口和拨号权限

如登录后需要重新获取拨号接口和拨号权限，使用 getextinterface.php 比使用 ctigetdevice.php 具有更高的效率。

- 播放录音

使用 getmonitorbyuid.php 接口实现转接操作根据呼叫编号获取录音文件。或者使用 getcdr.php 接口查询通话记录，然后使用 getmonitor.php 根据语音文件名称获取录音文件。

- 坐席状态

使用 getqueuemember.php 接口获取指定队列或坐席状态。

- 正在排队的电话
  使用 getqueuecalls.php 接口。
- 正在进行的通话
  使用 getcalls.php 接口。
- 获取用户（工号）列表
  使用 getusers.php 接口。
- 获取设备列表
  使用 getdevices.php 接口。

1.5.9 更多操作

WEBCTI 接口还提供短信管理、传真查询、语音留言、黑名单管理、电话会议、通话监听、资源监控、等功能，可参考《MVB2000_WEB 开发接口》。

2 设备状态监控

2.1 使用接口

getextensionstatus.php

2.2 说明

查看全部或某一设备的状态信息。若 url 中没有 exten 参数则会获取全部设备状态。

2.3 参数

说明	备注	必填
webkey	安全密钥	Y
exten	设备号	要监控的设备号（不使用此参数时为全部设备）
json	指定返回json变量(json=1/2/3)	2

2.4 URL

以监控设备 2011 为例：
http://192.168.0.201/webservice2/getextensionstatus.php?webkey=5b3c8c39f0ec17e1823e973c23fad009&exten=2011&json=2

2.5 结果

返回设备 2011 状态的 json 数据：
{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-01-30 11:30:35","exten":"2011","channel":"SIP/2011","context":"from-internal","status":"0","statustime":"2015-02-03 09:07:57"}]}

3 外拨

3.1 使用接口

ctioriginate.php

3.2 说明

假设用电话 7001 拨打电话 82754888，系统可以先接通 7001，7001 摘机后，再接通 82754888。也可以反过来先呼叫 82754888。

*此功能在 MVB2000 版本为 3.1.1.2 版本或以上时不会有话单和录音问题。

3.3 参数

3.3.1 先呼叫分机

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/7001(此参数为操作者的分机通道)
context	入口	from-internal
exten	号码	82754888（要拨打的号码）
priority	序号	1
application	应用	空
data	数据	空
timeout	超时	30000（此参数单位为毫秒）
callerid	主叫号码	4000820188 (来显号码)
variable	变量	FIRSTEXTCALLER=7001|MVB_CALLER=7001|MVB_CALLEE=82754888|MVB_DIRECT=OUT *此部分参数必须正确设置，否则会影响录音和话单
account	计费账号	空
async	同步	yes/no
json	指定返回json变量(json=1/2/3)	2

3.3.2 先呼叫外线

说明	备注	值
webkey	安全密钥	Y
channel	通道	Zap/g0/82754888 中继名称+外线号码
context	入口	from-internal
exten	号码	7001（内线号码）
priority	序号	1
application	应用	空
data	数据	空
timeout	超时	30000（此参数单位为毫秒）
callerid	主叫号码	4000820188 (来显号码)
variable	变量	FIRSTEXTCALLER=7001|MVB_CALLER=7001|MVB_CALLEE=82754888|MVB_DIRECT=OUT *此部分参数必须正确设置，否则会影响录音和话单
account	计费账号	空
async	同步	yes/no
json	指定返回json变量(json=1/2/3)	2

3.4 URL

3.4.1 先拨分机

http://192.168.0.103/webservice2/ctioriginate.php?webkey=0210f50063d2b950b2c6add6ea4b79a9&channel=SIP/7001&context=from-internal&exten=82754888&priority=1&callerid=4000820188&variable=FIRSTEXTCALLER=7620|MVB_CALLER=7620|MVB_CALLEE=82754888|MVB_DIRECT=OUT

主叫 被叫 来显 方向 录音
7620 82754888 4000820188 OUT 正常

3.4.2 先拨外线

http://192.168.0.103/webservice2/ctioriginate.php?webkey=0210f50063d2b950b2c6add6ea4b79a9&channel=Zap/g0/82754888&context=from-internal&exten=7620&priority=1&callerid=4000820188&variable=FIRSTEXTCALLER=7620|MVB_CALLER=7620|MVB_CALLEE=82754888|MVB_DIRECT=OUT

3.5 结果

返回的 json 数据：

{"errcode":0,"message":"Originate successfully queued","data":[]}

4 外拨播放 TTS 语音

4.1 使用接口

接口:ctioriginatetts.php

建立一个客户应用 ttsplay
exten=>s,1,NoOp()
;重复播放或提示选择次数,不设置默认 1
exten=>s,n,Set(TTS_PLAYCOUNT=3)
;接收按键输入的位数,不设置则不接收按键
exten=>s,n,Set(TTS_INPUTKEYLENGTH=1)
;重放按键,如按 1 键重放,上面的位数也对应设为 1
exten=>s,n,Set(TTS_REPLAYKEY=1)
exten=>s,n,AGI(ttsplayer.agi|base64)
exten=>s,n,NoOp(INPUT=${TTS_INPUTKEY})

TTS_INPUTKEY 通道变量包含用户输入的结果
ttsplayer.agi 为平台内置的程序，用户可以参照进行修改，实现更多功能。

4.2 说明

假设先呼叫 82754888，接通后播放“您好，畅信达通信欢迎您!”。

4.3 参数

调用 ctiorignatetts.php
先对“您好，畅信达通信欢迎您！”做 base64 编码,结果为：
xPq6w6Oss6nQxbTvzajQxbu2063E+g==

说明	备注	值
webkey	安全密钥	Y
channel	通道	Zap/g0/82754888 中继名称+外线号码
context	入口	custom-ttsplay
exten	号码	s
priority	序号	1
application	应用	空
data	数据	空
timeout	超时	30000（此参数单位为毫秒）

说明	备注	值
callerid	主叫号码	4000820188 (来显号码)
variable	变量	FIRSTEXTCALLER=7001|MVB_CALLER=7001|MVB_CALLEE=82754888|MVB_DIRECT=OUT|TTS_TEXT=xPq6w6Oss6nQxbTvzajQxbu2063E+g== *此部分参数必须正确设置，否则会影响录音和话单 *7001 为话单记录主叫号码，应根据需要设置 也可以在这里设置如下 3 个变量: TTS_PLAYCOUNT TTS_INPUTKEYLENGTH TTS_REPLAYKEY
account	计费账号	空
async	同步	yes/no
json	指定返回json变量(json=1/2/3)	2

4.4 URL

http://192.168.0.218/webservice2/ctioriginatetts.php
?webkey=0210f50063d2b950b2c6add6ea4b79a9
&channel=Zap/g0/82754888
&context= custom-ttsplay
&exten=s
&priority=1
&callerid=4000820188
&variable=FIRSTEXTCALLER=7001|MVB_CALLER=7001|MVB_CALLEE=82754888|MVB_DIRECT=OUT|TTS_TEXT= xPq6w6Oss6nQxbTvzajQxbu2063E+g==

4.5 结果

调用 ctioriginatetts.php 返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
```

<data/>
</root>

5 新版本强插

MVB2000 平台为 3.1.3.2 或以上版本可使用此功能

5.1 使用接口

ctioriginate.php

5.2 说明

要实现强插某一通话，监听者需调用 ctioriginate.php 向被强插方发起呼叫。接通后可以按键 456 切换监听、密语、强插模式。

5.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5002（强插者接口）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	ChanSpy
data	数据	5003|Bqsv(3) （被强插分机号|Bqsv(3) ）
timeout	超时	10000
callerid	主叫号码	5003(主叫号码)
variable	变量	空
account	计费账号	空
async	同步	no
json	指定返回json变量(json=1/2/3)	空

5.4 URL

http://192.168.0.103/webservice2/ctioriginate.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5002&application=ChanSpy&data=5003|Bqsv(3)&timeout=10000&callerid=5003&async=no

5.5 结果

返回 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

6 发起会议

6.1 使用接口

ctioriginate.php

6.2 说明

已 5002 发起会议为例，首先调用 ctioriginate.php 已分机 5002 创建会议室，然后再调用 ctioriginate.php 向其他分机发起会议呼叫。

6.3 参数

调用 ctioriginate.php 创建会议室：

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5002（操作者分机号码）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	MeetMe
data	数据	9111|pqd (会议室号码|pqd（会议室号码应动态生成，保证唯一）)
timeout	超时	10000
callerid	主叫号码	5002(操作者分机号码或其他)
variable	变量	空
account	计费账号	空
async	同步	空
json	指定返回json变量(json=1/2/3)	空

调用 ctioriginate.php 向其它分机发起会议呼叫，以向分机 5003 为例：

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5003（操作者分机号码）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	MeetMe

说明	备注	值
data	数据	9111|pq (会议室号码|pq（会议室号码应动态生成，保证唯一）)
timeout	超时	10000
callerid	主叫号码	5002(操作者分机号码或其他)
variable	变量	空
account	计费账号	空
async	同步	空
json	指定返回json变量(json=1/2/3)	空

6.4 URL

调用 ctioriginate.php 创建以分机 5002 创建会议室的 URL：
http://192.168.0.103/webservice2/ctioriginate.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5002&application=MeetMe&data=9111|pqd&timeout=10000&callerid=5002

调用 ctioriginate.php 向分机 5003 发起会议呼叫的 URL：
http://192.168.0.103/webservice2/ctioriginate.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5003&application=MeetMe&data=9111|pq&timeout=10000&callerid=5002

6.5 结果

调用 ctioriginate.php 创建以分机 5002 创建会议室的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

调用 ctioriginate.php 向分机 5003 发起会议呼叫的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

7 监听

MVB2000 平台为 3.1.3.2 或以上版本可在监听过程中按 5 可对座席耳语，按 6 可强插，按 4 恢复监听模式。

7.1 使用接口

ctioriginate.php

7.2 说明

要实现监听某一通话，监听者需调用 ctioriginate.php 向被监听方发起呼叫。

7.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5002（监听者接口）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	ChanSpy
data	数据	5003|dqsv(3)（被监听者分机号|dqsv(3) ）
timeout	超时	10000
callerid	主叫号码	5003(主叫号码)
variable	变量	空
account	计费账号	空
async	同步	no
json	指定返回json变量(json=1/2/3)	空

7.4 URL

http://192.168.0.103/webservice2/ctioriginate.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5002&application=ChanSpy&data=5003|dqsv(3)&timeout=10000&callerid=5003&async=no

7.5 结果

返回 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

8 三方通话

8.1 使用接口

ctioriginate.php
ctiredirect.php
cticommand.php

8.2 说明

首先要在 MVB2000 系统的客户应用里添加应用，此应用用来建立动态会议室：
客户应用里增加一个名称为 **dynamicmeetme** 的应用，内容如下：

```
exten => _X.,1,Answer
exten => _X.,n,NoOp(EXTEN=${EXTEN})
exten => _X.,n,MeetMe(${EXTEN}|dq)
exten => _X.,n,Macro(hangupcall)
exten => h,1,Hangup
```

假设 8003 与 5003 正在通话，第三方为 5002
调用 ctioriginate.php 接口向第三方 5002 发起呼叫，建立动态会议室将 5002 放入会议室，调用 ctiredirect.php 接口将正在通话的 8003 和 5003 放入会议室中，最后调用 cticommand.php 接口监控会议。

8.3 参数

调用 ctiorignate.php

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5002（操作者分机号码）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	MeetMe
data	数据	9111|pqd (会议室号码|pqd（会议室号码应动态生成，保证唯一）)
timeout	超时	10000
callerid	主叫号码	5002(操作者分机号码或其他)
variable	变量	空
account	计费账号	空
async	同步	空

说明	备注	值
json	指定返回json变量(json=1/2/3)	空

调用 ctiredirect.php：

说明	备注	值
webkey	安全密钥	Y
channel	通道	IAX2/iax100-1067（主叫通道）
extrachannel	附加通道	SIP/5003-0a08d998（被叫通道）
context	入口	custom-dynamicmeetme
exten	号码	9111（调用ctioriginate,php时创建的会议室号码）
priority	序号	1
json	指定返回json变量(json=1/2/3)	空

注：此参数中的主叫通道和被叫通道可用 getchannel.php 获取，参数为分机对应的 channel 参数。例如 8003 拨打给 5003，那么参数 callerid 为 8003 所对应的参数 channel 即为主叫通道，参数 callerid 为 5003 所对应的参数 channel 即为被叫通道。

调用 cticommand.php：

说明	备注	值
webkey	安全密钥	Y
command	命令	meetme list 9111（调用ctioriginate,php时创建的会议室号码）
json	指定返回json变量(json=1/2/3)	空

8.4 URL

调用 ctioriginate.php 的 URL：
http://192.168.0.103/webservice2/ctioriginate.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5002&application=MeetMe&data=9111|pqd&timeout=10000&callerid=5002
调用 ctiredirect.php 的 URL：
http://192.168.0.103/webservice2/ctiredirect.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=IAX2/iax100-1067&extrachannel=SIP/5003-0a08d998&context=custom-dynamicmeetme&exten=9111&priority=1
调用 cticommand.php 的 URL：
http://192.168.0.103/webservice2/cticommand.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&command=meetme%20list%209111

8.5 结果

调用 ctioriginate.php 返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

调用 ctiredirect.php 返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Dual Redirect successful</message>
<data/>
</root>
```

调用 cticommand.php 返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>SUCCESS</message>
<data>
<item>
<result>
Privilege: Command ActionID: WEB1252205885 User #: 05 5003 <noname> Channel: SIP/5003-0a08d998 (unmonitored) User #: 06 5002 <noname> Channel: SIP/5002-0a0a4fd8 (unmonitored) User #: 07 8003 苏 文 佳 Channel: IAX2/iax100-11815 (unmonitored) User #: 08 5003 <noname> Channel: SIP/5003-0a0720c0 (unmonitored) 4 users in that conference.
</result>
</item>
</data>
</root>
```

9 拆线

9.1 使用接口

ctihangup.php

9.2 说明

将正在通话中的线路拆除

9.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5003-0a11f460（要拆除的线路通道）
json	指定返回json变量(json=1/2/3)	2

注：注：此参数中的主叫通道和被叫通道可用 getchannel.php 获取，参数为分机对应的 channel 参数。例如分机 5003，那么参数 callerid 为 5003 所对应的参数 channel 即为通道。

9.4 URL

http://192.168.0.103/webservice2/ctihangup.php?webkey=33de2de41e1fe862e6b0e0500bc515cd&channel=SIP/5003-0a11f460&json=2

9.5 结果

返回的 json 数据：

{"errcode":0,"message":"Channel Hungup","data":[]}

10 获取座席状态

10.1 使用接口

getextensionstatus.php
getqueuemember.php

10.2 说明

获取座席状态包括获取座席分机状态和获取队列成员状态。
以座席 5005 为例。
调用 getqueuemember.php 时若 url 中没有 queue 参数则会获取全部队列成员状态。

10.3 参数

调用 getextensionstatus.php 获取座席分机状态：

说明	备注	值
webkey	安全密钥	Y
exten	分机号	2011 （座席分机号）
json	指定返回json变量(json=1/2/3)	2

调用 getqueuemember.php 获取队列成员状态：

说明	备注	值
webkey	安全密钥	Y
queue	队列号	90 （座席所在队列号）
agentname	座席号	2011 （座席号）
json	指定返回json变量(json=1/2/3)	2

10.4 URL

调用 getextensionstatus.php 获取座席分机状态的 URL：
http://192.168.0.201/webservice2/getextensionstatus.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&exten=2011&json=2
调用 getqueuemember.php 获取队列成员状态的 URL：
http://192.168.0.201/webservice2/getqueuemember.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=90&agentname=2011&json=2

10.5 结果

调用 getextensionstatus.php 获取座席分机状态返回的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-01-19 10:59:38","exten":"2011","channel":"SIP/2011","context":"from-internal","status":"1","statustime":"2015-01-21 09:32:27"}]}

调用 getqueuemember.php 获取队列成员状态返回的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-01-23 10:21:19","queue":"90","agent":"SIP/2011","agentname":"2011","agentship":"dynamic","penalty":"0","callstaken":"0","lastcall":"0000-00-00 00:00:00","status":"1","paused":"0"}]}

11 获取队列状态

11.1 使用接口

getqueue.php

11.2 说明

以获取 90 队列为例。
若 url 中没有 queue 参数则会获取全部队列状态

11.3 参数

说明	备注	值
webkey	安全密钥	Y
queue	队列号	90
json	指定返回json变量(json=1/2/3)	2

11.4 URL

http://192.168.0.103/webservice2/getqueue.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=90&json=2

11.5 结果

返回的 json 数据：
{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-01-23 09:29:16","queue":"90","max":"6","calls":"0","holdtime":"0","completed":"0","abandoned":"0","serviceLevel":"0","servicelevelperf":"0.0","weight":"0"}]}

12 获取录音

12.1 使用接口

getmonitor.php

12.2 说明

假设要获取的录音文件名为 OUT5050-20090806-172454-1249550694.0.WAV。

12.3 参数

说明	备注	值
webkey	安全密钥	Y
filename	文件名	
mp3	转换mp3格式，参数值为1或yes	

12.4 URL

直接下载 WAV 格式录音文件
http://192.168.0.103/webservice2/getmonitor.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&filename=OUT5050-20090806-172454-1249550694.0.WAV
下载录音文件，并将文件转换为 MP3 格式
http://192.168.0.103/webservice2/getmonitor.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&filename=OUT5050-20090806-172454-1249550694.0.WAV&mp3=yes

12.5 结果

用户可打开录音文件听取录音或将录音文件下载。

13 示忙/示闲

13.1 使用接口

ctiqueuepause.php

13.2 说明

以座席 5003 为例，分机号为 5003，所在队列为 5090

13.3 参数

说明	备注	值
webkey	安全密钥	Y
queue	队列号	5090
interface	座席接口	SIP/5003
paused	状态（true/false，1/0）	true/false
json	指定返回json变量(json=1/2/3)	2

13.4 URL

示忙：
http://192.168.0.103/webservice2/ctiqueuepause.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=5090&interface=SIP/5003&paused=true&json=2
示闲：
http://192.168.0.103/webservice2/ctiqueuepause.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=5090&interface=SIP/5003&paused=false&json=2

13.5 结果

示忙：
{"errcode":0,"message":"Interface paused successfully","data":[]}
示闲：
{"errcode":0,"message":"Interface unpaused successfully","data":[]}

14 登入队列

14.1 使用接口

ctiqueueadd.php

14.2 说明

以座席 5003 为例，加入到 5090 队列。

14.3 参数

说明	备注	值
webkey	安全密钥	Y
queue	队列号	5090
interface	座席接口	SIP/5003
penalty	等级	空
json	指定返回json变量(json=1/2/3)	空

14.4 URL

http://192.168.0.103/webservice2/ctiqueueadd.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=5090&interface=SIP/5003

14.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Added interface to queue</message>
<data/>
</root>
```

15 登出队列

15.1 使用接口

ctiqueueremove.php

15.2 说明

以座席 5003 为例，从 5090 队列中登出。

15.3 参数

说明	备注	值
webkey	安全密钥	Y
queue	队列号	5090
interface	座席接口	SIP/5003
json	指定返回json变量(json=1/2/3)	空

15.4 URL

http://192.168.0.103/webservice2/ctiqueueremove.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&queue=5090&interface=SIP/5003

15.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Removed interface from queue</message>
<data/>
</root>
```

16 获取通道变量

16.1 使用接口

ctigetvar.php

16.2 说明

要调用此接口前必须确保通道被创建，而通道只有在双方建立通话后才创建。
以 8003 与 5003 建立通话为例，5003 通道为 SIP/5003-08cb5938，获取此通道的 DIALEDPEERNUMBER 变量。
8003 与 5003 建立通话后，可调用 getchannel.php 查看通道

*可以使用
ctigetchanneldetail.php 批量获取常用通道参数
ctigetvars.php 批量获取多个指定通道变量

16.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5003-08cb5938
variable	变量名称	DIALEDPEERNUMBER
json	指定返回json变量(json=1/2/3)	2

16.4 URL

http://192.168.0.103/webservice2/ctigetvar.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&channel=SIP/5003-08cb5938&variable=DIALEDPEERNUMBER&json=2

16.5 结果

返回的 json 数据：
{"errcode":0,"message":"SUCCESS","data":[{"result":"5003"}]}

17 设置通道变量

17.1 使用接口

ctisetvar.php

17.2 说明

要调用此接口前必须确保通道被创建，而通道只有在双方建立通话后才创建。
以 8003 与 5003 建立通话为例，5003 通道为 SIP/5003-08cb5938，设置此通道的 CALLERID 变量为 5003。
8003 与 5003 建立通话后，可调用 getchannel.php 查看通道。

17.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5003-08cb5938
variable	变量名称	CALLERID
value	值	5003
json	指定返回json变量(json=1/2/3)	2

17.4 URL

http://192.168.0.103/webservice2/ctisetvar.php?webkey=7fc80a51d24676ca8354a3a3dfc718f4&channel=SIP/5003-08cb5938&variable=CALLERID&value=5003&json=2

17.5 结果

返回的 json 数据：

{"errcode":0,"message":"Variable Set","data":[]}

18 获取弹屏数据

18.1 使用接口

getpopupscreen.php

18.2 说明

假设座席分机为 2011，当电话 2012 拨打给 2011 或 2011 拨打给 2012，在振铃时调用此接口，可获取相关弹屏数据。获取弹屏数据后如何获得更多的信息，请参考“批量获取常用变量”。

18.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/2011
json	指定返回json变量(json=1/2/3)	2

18.4 URL

http://192.168.0.103/webservice2/getpopupscreen.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=SIP/2011&json=2

18.5 结果

createtime	日期时间
channel	活动通道
uniqueid	呼叫编号
callerid	来电号码
calleridname	来电名称
direct	呼叫方向(IN/OUT/空)
miscdata	附加数据(通常是关联通道)
miscuniqueid	附加数据(通常是关联编号)
state	状态(Ring/Ringing/Up/Down)
did	原被叫号码

返回 8003 拨 5003 时的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-02-04 09:59:22","channel":"SIP/2011-0894c548","uniqueid":"1423015162.92","callerid":"2012","calleridname":"???
","direct":"IN","miscdata":"SIP/2012-b7c36250","miscuniqueid":"1423015162.91","state":"Up","did":""}]}

返回 5003 拨 8003 时的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"createtime":"2015-02-04 09:59:58","channel":"SIP/2011-b7c36250","uniqueid":"1423015198.93","callerid":"2012","calleridname":"","direct":"OUT","miscdata":"SIP/2012-0894c548","miscuniqueid":"1423015198.94","state":"Up","did":""}]}

XML 图示：

19 批量获取常用变量

19.1 使用接口

ctigetchanneldetail.php

19.2 说明

获取弹屏数据后，需要继续获取被叫号码、队列号码、IVR 历史、录音文件名称等变量，可以调用此接口一次获得。由于此类变量附加在主叫通道，所以当您传入的通道不是主叫通道时，需要设置 type=1 或 2，并设置超时时间，在指定的超时时间内主叫和被叫通道接通时，接口返回。当传入的变量是主叫通道，则不需要设置超时，type=0，接口立刻返回。

19.3 参数

说明	备注	必填
webkey	安全密钥	Y
channel	活动通道	Y
type	类型:0/1/2 0-获取当前通道的变量，1-获取关联通道的变量（需要通话建立），2-自动判断（总是获取主叫通道的变量，需要通话建立） 默认：0	N
timeout	等待通道桥接时间（秒），默认：0，最大：120。设置此参数后，等待系统设置“BRIDGEPEER”变量，此变量有值说明主被叫通道已经建立通话，所以当您需要在通话建立时获取变量，可以设置此超时值。	N
json	指定返回json变量(json=1/2/3)	N

19.4 URL

http://192.168.0.102/webservice2/ctigetchanneldetail.php?webkey=1b709663c6c05f23c08f4b9388ac1a7d&channel=SIP/7001-0a8ce848&type=0 &json=2

19.5 结果

channel	活动通道
uniqueid	呼叫编号
dstchannel	活动通道关联通道
datachannel	变量获取通道

dstuniqueid	关联编号
did	原被叫号码
monitorfile	录音文件名（无扩展名）
queuenum	队列号
agentno	坐席号
ivrhist	IVR历史（IVR编号，用-分隔）
holdtime	等待时间（秒）
caller	主叫号码
callee	被叫号码
areacode	地区号码
city	城市
province	省份
simcardtype	电话卡类型

返回的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"channel":"SIP/7001-0a8ce848","uniqueid":"1421274641.0","dstchannel":"SIP/100-102-0a8ba5a0","datachannel":"SIP/7001-0a8ce848","dstuniqueid":"1421274641.0","did":"","monitorfile":"","dialednum":"2012","queuenum":"","agentno":"","ivrhist":"","holdtime":"","caller":"2011","callee":"2012","areacode":"","city":"","province":"","simcardtype":""}]}

20 质检

20.1 使用接口

getcalls.php
ctisetvar.php
ctiredirect.php

20.2 说明

假设座席号码为 5003，客户号码为 8003，质检客户。在调用 ctisetvar.php 用到的参数中：
被转接通道(客户)：可以通过 getcalls.php 获得。
呼叫编号：可以使用弹屏时获得的数据（miscuniqueid），也可以可以通过 getcalls.php 获得。

质检过程是先调用 ctisetvar.php 设置变量，再调用 ctiredirect.php 将客户通道转接到质检应用中。

20.3 参数

调用 getcalls.php：

说明	备注	值
webkey	安全密钥	Y
srcchannel	主叫通道	SIP/5003
dstchannel	被叫通道	IAX2/iax100

获取参数 dstchannel（被转接通道）和 dstuniqueid（呼叫编号）。

调用 ctisetvar.php 要设置的变量：

说明	备注	值
webkey	安全密钥	Y
channel	被转接通道(客户)	IAX2/iax100-16385
variable	要设置的变量名称	MVB_QMCALLERID
value	来电号码	5003

说明	备注	值
webkey	安全密钥	Y
channel	被转接通道(客户)	IAX2/iax100-16385
variable	要设置的变量名称	MVB_QMAGENTNO
value	座席号码（分机号）	5003

说明	备注	值
webkey	安全密钥	Y
channel	被转接通道(客户)	IAX2/iax100-16385
variable	要设置的变量名称	MVB_QMUNIQUEID(图示中的对方呼叫编号)
value	呼叫编号	1249611782.110

调用 ctiredirect.php 的参数：

说明	备注	值
webkey	安全密钥	Y
channel	被转接通道(客户)	IAX2/iax100-16385
extrachannel	附加通道	空
context	入口	custom-advqm
exten	号码	s（注意大小写）
priority	序号	1

20.4 URL

调用 getcalls.php 的 URL：

http://192.168.0.103/webservice2/getcalls.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&srcchannel=SIP/5003&dstchannel=IAX2/iax100

调用 ctisetvar.php 设置变量 MVB_QMCALLERID 的 URL：

http://192.168.0.103/webservice2/ctisetvar.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=IAX2/iax100-16385&variable=MVB_QMCALLERID&value=5003

调用 ctisetvar.php 设置变量 MVB_QMAGENTNO 的 URL：

http://192.168.0.103/webservice2/ctisetvar.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=IAX2/iax100-16385&variable=MVB_QMAGENTNO&value=5003

调用 ctisetvar.php 设置变量 MVB_QMUNIQUEID 的 URL：

http://192.168.0.103/webservice2/ctisetvar.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=IAX2/iax100-16385&variable=MVB_QMUNIQUEID&value=1249611782.110

调用 ctiredirect.php 的 URL：

http://192.168.0.103/webservice2/ctiredirect.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=IAX2/iax100-16385&context=custom-advqm&exten=s&priority=1

20.5 结果

调用 getcalls.php 的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>SUCCESS</message>
<data>
<item>
<createtime>2015-02-04 10:11:30</createtime>
<srcchannel>SIP/2011-b7c36250</srcchannel>
<dstchannel>SIP/2012-0894c548</dstchannel>
<callstate>Link</callstate>
<srccallerid>2011</srccallerid>
<srccalleridname>???¨ </srccalleridname>
<dstcallerid>2012</dstcallerid>
<dstcalleridname><Unknown></dstcalleridname>
<srcchannelstate>Up</srcchannelstate>
<dstchannelstate>Up</dstchannelstate>
<srcuniqueid>1423015881.95</srcuniqueid>
<dstuniqueid>1423015881.96</dstuniqueid>
<dialtime>2015-02-04 10:11:21</dialtime>
<linktime>2015-02-04 10:11:30</linktime>
<unlinktime>0000-00-00 00:00:00</unlinktime>
<clearflag>0</clearflag>
</item>
</data>
</root>
```

调用 ctisetvar.php 设置变量 MVB_QMCALLERID 的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Variable Set</message>
<data/>
</root>
```

调用 ctisetvar.php 设置变量 MVB_QMAGENTNO 的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Variable Set</message>
<data/>
</root>
```

调用 ctisetvar.php 设置变量 MVB_QMUNIQUEID 的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Variable Set</message>
<data/>
</root>
```

调用 ctiredirect.php 的 XML 数据：

```
<root>
<errcode>0</errcode>
<message>Redirect successful</message>
<data/>
</root>
```

21 在线获取用户按键输入

21.1 使用接口

ctiwaitauth.php

21.2 说明

在通话过程中坐席可以在电话上按快捷键或者通过调用接口在电脑上点击鼠标实现让客户输入按键选择，输入完成后界面可以显示输入结果，通话自动恢复。

21.3 参数

说明	备注	必填
webkey	安全密钥	Y
channel	通道	Y
code	快捷键编码	需要其中一项，code优先
name	快捷键名称	
type	指定获取变量的通道。 1--本通道 0-对方通道，默认0	N
timeout	等待结果时间，单位秒，默认：120。0秒为不等待立即返回。	N
json	指定返回json变量(json=1/2/3)	N

21.4 URL

http://192.168.0.102/webservice2/ctiwaitauth.php?webkey=1b709663c6c05f23c08f4b9388ac1a7d&channel=SIP/7001-0a3cc5d0&code=06&type=0&timeout=20&json=2

21.5 结果

返回的 json 数据：

{"errcode":0,"message":"SUCCESS","data":[{"CALLDATA_STATE":"1","CALLDATA_ID":"(null)","CALLDATA_ACCOUNT":"(null)","CALLDATA_AUTH":0,"CALLDATA_TIMESTAMP":"1324864355","CALLDATA_DATA1":"4","CALLDATA_DATA2":"(null)","CALLDATA_DATA3":"(null)","CALLDATA_DATA4":"(null)"}]}

返回:

CALLDATA_ID 客户号或会员id号

CALLDATA_ACCOUNT	账号或卡号
CALLDATA_AUTH	认证状态　0-未处理 1-通过 2-失败
CALLDATA_TIMESTAMP	数据生成时间戳;
CALLDATA_STATE	程序运行状态　0=未处理 1=完成 2-输入不完整 3-挂机
CALLDATA_DATA1	程序返回数据1（如按键结果）
CALLDATA_DATA2	程序返回数据2
CALLDATA_DATA3	程序返回数据3
CALLDATA_DATA4	程序返回数据4

以上数据由快捷键调用的AGI程序生成。

21.6 实现过程

场景：8001 号码通过外线呼入进入队列后接通坐席 7000 或直接接通坐席 7000。坐席 7000 对应的设备接口是 SIP/7001(可以预定义、也可以使用接口 getextinterface.php 动态获取坐席对应的设备接口)。

1. 定义接收用户按键的快捷键

在 mvb2000 设置→快捷键菜单增加一个快捷键，如下图

其中：wait_option.agi 为平台内置的接收用户输入的语音接口程序，简单的应用可以直接使用，复杂的需求可以自己编制这个接口或者委托我们编制。

wait_option.agi 后面的内容位参数，全部用|分隔，参数数量固定，不设置参数值也需要有|分隔,如||。参数作用如下：

第一个：提示输入的语音，可以从系统录音上传，然后在这里输入“custom/”加录音名称。

第二个：输入完成后的提示语音，可以从系统录音上传，然后在这里输入“custom/”加录音名称，不定义不播放。

第三个：按键输入长度，比如 1，用户输入一位后即返回。当输入长度匹配此设置时即终止输入，也可以按#终止输入。如：长度为 10，用户输入 123#，系统接收到 123，同时终止输入。

第四个：有效按键。如：1234，用户只能输入 1234 中的一个数字。0123456789*表示全部有效。#为回车键（输入结束键）。
第五个：允许超时或输入错误次数。如：3，最多允许超时提示 3 次。
第六个：提示完后，等待输入的时间（秒），如：10，提示完后，10 秒内不输入则再次提示，同时超时次数加一。
如：wait_option.agi|custom/welcome|custom/thankyou|1|0123456789|3|10
ctiwaitauth.php 中调用时可以使用名称或快捷键。

2. 每隔 2 秒调用弹屏接口获取来电信息和活动通道，当来电时获取到如下信息：

http://192.168.0.102/webservice/getpopupscreen.php?webkey=a9d92aa9b60375919b9029250c7db841&channel=SIP/7001

```
▼<root>
   <errcode>0</errcode>
   <message>SUCCESS</message>
  ▼<data>
    ▼<item>
       <createtime>2015-02-04 09:59:58</createtime>
       <channel>SIP/2011-b7c36250</channel>   坐席活动通道
       <uniqueid>1423015198.93</uniqueid>   坐席呼叫编号
       <calleid>2012</calleid>   来电号码
       <calleridname/>
       <direct>OUT</direct>
       <miscdata>SIP/2012-0894c548</miscdata>   对方活动通道
       <miscuniqueid>1423015198.94</miscuniqueid>   对方呼叫编号
       <state>Up</state>
       <did/>
     </item>
   </data>
 </root>
```

http://192.168.0.102/webservice/ctiwaitauth.php?webkey=a9d92aa9b60375919b9029250c7db841&channel=SIP/7001-0a95a118&code=06&type=0&timeout=30

3. 调用 ctiwaitauth.php 接口获取按键输入
code=06：使用前面定义的快捷键 06。
channel=SIP/7001-0a95a118:活动通道使用弹屏接口获取的坐席活动通道，根据快捷键定义的提示类型，会在对方通道提示输入。
type=0：在对方活动通道（即 SIP/100-102-0a949e98）上获取变量。
timeout=30：等待 30 秒。
调用后用户会听到参数 1 定义的语音，输入 1 个按键后，会听到参数 2 对应的语音，然后接口返回结果。
输入结果如下：
CALLDATA_STATE:1　　用户输入结束（0=未处理 1=完成 2-输入不完整 3-挂机）
CALLDATA_DATA1:8　　按键为 8。
这里的 CALLDATA_*变量为改程序中定义的通道变量，可根据需要进行赋值。
在 wait_option.agi 中，CALLDATA_STATE 表示输入状态，CALLDATA_DATA1 存放按键。

22 会议监控

22.1 使用接口

cticommand.php

22.2 说明

假设分机 5003 建立会议室，会议成员还有分机 8003 和 5002，会议室编号为19465983。执行监控后当会议室只有一个成员时，关闭会议。防止线路始终挂起。

22.3 参数

说明	备注	值
webkey	安全密钥	Y
command	命令	meetme list 19465983 （meetme list 要监控的会议室号码）
json	指定返回json变量(json=1/2/3)	空

22.4 URL

http://192.168.0.103/webservice2/cticommand.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&command=meetme%20list%2019465983

22.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>SUCCESS</message>
<data>
<item>
<result>
Privilege: Command ActionID: WEB1273720988 User #: 01 5002 <noname> Channel: SIP/5002-09598800 (unmonitored) User #: 02 8003 <noname> Channel: IAX2/iax100-20200 (unmonitored) 2 users in that conference.
</result>
</item>
```

```
</data>
</root>
```

23 转接

23.1 使用接口

ctiredirect.php

23.2 说明

假设分机 8003 拨 5003，接通后，分机 5003 将 8003 电话转接给 5002。

23.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	IAX2/iax100-1538
extrachannel	附加通道	空
context	入口	from-internal
exten	号码	5002
priority	序号	1
json	指定返回json变量(json=1/2/3)	空

注：此参数中的主叫通道和被叫通道可用 getchannel.php 获取，参数为分机对应的 channel 参数。例如 8003 拨打给 5003，那么参数 callerid 为 8003 所对应的参数 channel 即为通道。

23.4 URL

已监控分机 5013 为例：
http://192.168.0.103/webservice2/ctiredirect.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=IAX2/iax100-1538&context=from-internal&exten=5002&priority=1

23.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Redirect successful</message>
<data/>
</root>
```

24 听留言

24.1 使用接口

ctioriginate.php

24.2 说明

假设听取分机 8003 的留言，调用 ctiorignate.php，分机 8003 摘机后即可听取留言。

24.3 参数

调用 ctiorignate.php：

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5003（要听取留言的分机号码）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	VoiceMailMain
data	数据	5003@default|s
timeout	超时	10000
callerid	主叫号码	5003 (要听取留言的分机号码或其他)
variable	变量	空
account	计费账号	空
async	同步	空
json	指定返回json变量 (json=1/2/3)	空

24.4 URL

http://192.168.0.103/webservice2/ctioriginate.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=SIP/5003&application=VoiceMailMain&data=5003@default|s&callerid=5003

24.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

25 获取留言数

25.1 使用接口

ctimailboxcount.php

25.2 说明

以分机 5003 为例，获取 5003 的语音留言数。获取的信息中 NewMessages 为新留言数，OldMessages 为旧留言数。

25.3 参数

说明	备注	值
webkey	安全密钥	Y
mailbox	信箱号码	5003
json	指定返回json变量(json=1/2/3)	2

25.4 URL

http://192.168.0.103/webservice2/ctimailboxcount.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&mailbox=5003&json=2

25.5 结果

返回的 json 数据：

{"errcode":0,"message":"Mailbox Message Count","data":[{"mailbox":"5003","oldmessages":"0","newmessages":"0"}]}

26 代接

26.1 使用接口

ctioriginate.php

26.2 说明

以 8003 拨给 5003 为例，分机 5002 代接。

26.3 参数

说明	备注	值
webkey	安全密钥	Y
channel	通道	SIP/5002（要代接的分机号码）
context	入口	空
exten	号码	空
priority	序号	空
application	应用	PickUp
data	数据	5003 （背代接的分机号）
timeout	超时	10000
callerid	主叫号码	8003
variable	变量	空
account	计费账号	空
async	同步	空
json	指定返回json变量 (json=1/2/3)	空

注意：如果被代接与代接分机不在一个组中，则以上的 data 参数应写为 5003@from-internal，其中 from-internal 为通过查询得到的 5003 分机入口。

26.4 URL

http://192.168.0.103/webservice2/ctioriginate.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&channel=SIP/5002&application=PickUp&data=5003&timeout=10000&callerid=8003

26.5 结果

返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>Originate successfully queued</message>
<data/>
</root>
```

27 会议控制

27.1 使用接口

cticommand.php

27.2 说明

对会议的控制添加人员，踢出人员，会议室静音，会议室加锁/解锁，关闭会议室等。
会议控制通过调用 cticommand.php 接口运行 meetme 命令来实现的，其命令写法如下：

meetme (un)lock|(un)mute|kick|list <confno> <usernumber>

参数：
lock：会议加锁。
unlock：会议解锁。
mute：静音。
unmute：取消静音。
kick：踢出人员。
list：列出会议室信息（若要查询所有会议室编号等信息可用此命令）。

confno：为要操作的会议室编号。
usernumber：为要操作的会议室成员编号。
假设会议室编号 36405501，成员有分机 8003 和 5003，5003 为创建者。
以将 8003 踢出会议室为例，命令写法如下：
meetme kick 36405501 02

27.3 参数

调用 cticommand.php：

说明	备注	值
webkey	安全密钥	Y
command	命令	meetme kick 36405501 02
json	指定返回json变量(json=1/2/3)	空

27.4 URL

调用 cticommand.php 的 URL：
http://192.168.0.103/webservice2/cticommand.php?webkey=a1e9ddf61058ba180bd3499dcf300cbc&command=meetme%20kick%2036405501%2002

27.5 结果

调用 cticommand.php 返回的 xml 数据：

```
<root>
<errcode>0</errcode>
<message>SUCCESS</message>
<data>
<item>
<result>
Privilege: Command ActionID: WEB1249630556
</result>
</item>
</data>
</root>
```

28 获取指定目录录音/留言文件

28.1 使用接口

getspoolfile.php

28.2 说明

对录音文件和留言文件的控制包括打开、下载和删除。
以打开/下载名为 v_4006618718-8005-20090714-114206-1247542918.7.WAV 的录音文件为例。

28.3 参数

调用 getspoolfile.php：

说明	备注	值
webkey	安全密钥	Y
location	存放录音/留言的最终目录	mvb2k400_voicemsg（此参数为存放录音/留言文件的最终根目录的文件夹名称）
filename	录音/留言文件的名字	v_4006618718-8005-20090714-114206-1247542918.7.WAV （注意加文件扩展名）
delete	对文件的操作	yes/no ,1/0(yes/1 为删除录音/留言文件，no/0 为打开/下载录音/留言文件)

28.4 URL

打开/下载录音文件的 URL：
http://192.168.0.103/webservice2/getspoolfile.php?webkey=ac6160728980fd2fda2993a5b405da50&location=mvb2k400_voicemsg&filename=v_4006618718-8005-20090714-114206-1247542918.7.WAV&delete=no

28.5 结果

接口调用成功后，将弹出文件下载对话框，可选择打开或下载文件。

29 弹屏时如何获得用户按键历史、队列号码、等待时间、原被叫号码？

使用 **ctigetvar.php** 接口获取主叫通道上的如下变量：

MVB_IVRHIST	IVR 按键历史
MVB_QUEUENAME	队列号码
MVB_HOLDTIME	等待时间
MVB_DNID	原被叫
MVB_ANI	原主叫
UNIQUEID	呼叫编号

也可以获取其他变量，详见《MVB2000_WEB 开发接口(v2.0.0.0).doc》第 6.1 部分“通道变量”说明。

也可以使用批量获取变量接口，一次获取多个变量。

如：

http://192.168.0.100/webservice2/ctigetvars.php?webkey=123caaafe6cf489faeb2d6a4b8c5e554&channel=Zap/8-1&variable0=MVB_DNID&variable1=MVB_ANI&variable2=MVB_QUEUENAME&variable3=QUEUEAGENTNO&variable4=MVB_IVRHIST

30 如何集成登录功能

30.1 在 MVB2000 平台设置 WEB 密钥

在 MVB2000 设置-->[基本设置]界面设置您需要的 WEB 密钥，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。

30.2 webkey 计算

WEBCTI 接口和集成登录中用到的 webkey 参数均使用此方法计算。由于计算中用到了日期要素，所以必须保证您使用的日期与平台日期一致。

webkey 使用 mvb2000 基本设置里的 web 密钥字符串+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。如：

mvb2000 平台的[WEB 密钥]为“0866089e569bbd3d”，当日日期为 2011-06-21，则组合字符串为“0866089e569bbd3d2011-06-21”，md5 结果为:“ffccb30a388e48ade2d7ea56909cbabf”。

30.3 集成登录

1、开启集成登录功能： 在 MVB2000 设置-->[基本设置]界面开启集成登录功能。

2、在 MVB2000 设置-->[管理员设置]界面建立集成登录使用的管理员和密码，并配置相应权限。

3、通过 HTTP POST 方式登录平台，跳转到您需要集成的页面。POST 参数如下：

webkey 根据上述方法或者根据《MVB2000_WEB 开发接口》的 webkey 设置部分计算出的密钥参数。

frienduser 用户名，管理员设置中建立的用户名。

friendpass 密码，管理员设置中建立的用户密码经 md5 加密后的值。如：123456 加密后的值是“e10adc3949ba59abbe56e057f20f883e”。

form 跳转 action 可以是:

http://192.168.0.218/admin/config.php,

http://192.168.0.218/admin/config.php?type=setup&display=ivr ,

http://192.168.0.218/admin/config.php?type=tool ,

http://192.168.0.218/admin/reports.php 等。

您可以根据 MVB2000 平台中个模块的链接地址任意设定。

30.4 集成登录 PHP 样例

```
<?php
$webkey='0966089e569bcd3d';
$newkey=md5($webkey.date('Y-m-d'));
?>
<html>
<body>
<form name='mvb2000auth' action="http://192.168.0.218/admin/config.php"
method='post'>
<input type="hidden" name='webkey' value="<?php echo $newkey; ?>">
<input type="hidden" name='frienduser' value="admin">
<input type="hidden" name='friendpass' value="<?php echo md5('123456');?>">
<input type="submit" value="登录">
</form>
</body>
</html>
```

30.5 C#计算 MD5 方法

```
/*MD5
s=”123456”
结果: e10adc3949ba59abbe56e057f20f883e
*/
private string get_md5(string s)
{
    string strResult = "";
    MD5 md5 = MD5.Create();
    byte[] bytResult = md5.ComputeHash(System.Text.Encoding.ASCII.GetBytes(s));
    for (int i = 0; i < bytResult.Length; i++)
     {
          string hexOutput = String.Format("{0:x2}", bytResult[i]);
          strResult = strResult + hexOutput;
      }
      md5.Clear();
      return strResult;
  }
/*按接口要求计算WEBKEY
mvbwebkey =”0866089e569bbd3d”，当日日期为2011-06-21
则组合字符串为”0866089e569bbd3d2011-06-21”
结果为：”ffccb30a388e48ade2d7ea56909cbabf”
```

```
*/
   private string get_webkey(mvbwebkey)
    {
        string strResult="";
        string mvbwebkey = "";
        DateTime currentTime = DateTime.Now;
        if (mvbwebkey.Length == 0)    return null;
        mvbwebkey = mvbwebkey + currentTime.ToString("yyyy-MM-dd");
        strResult = get_md5(mvbwebkey);
        return strResult;
     }
```

30.6　javascript 计算 MD5 方法

```
/*js MD5 类*/
function MD5(sMessage) {
	function RotateLeft(lValue, iShiftBits) { return (lValue<<iShiftBits) | (lValue>>>(32-iShiftBits)); }
	function AddUnsigned(lX,lY) {
		var lX4,lY4,lX8,lY8,lResult;
		lX8 = (lX & 0x80000000);
		lY8 = (lY & 0x80000000);
		lX4 = (lX & 0x40000000);
		lY4 = (lY & 0x40000000);
		lResult = (lX & 0x3FFFFFFF)+(lY & 0x3FFFFFFF);
		if (lX4 & lY4) return (lResult ^ 0x80000000 ^ lX8 ^ lY8);
		if (lX4 | lY4) {
			if (lResult & 0x40000000) return (lResult ^ 0xC0000000 ^ lX8 ^ lY8);
			else return (lResult ^ 0x40000000 ^ lX8 ^ lY8);
		} else return (lResult ^ lX8 ^ lY8);
	}
	function F(x,y,z) { return (x & y) | ((~x) & z); }
	function G(x,y,z) { return (x & z) | (y & (~z)); }
	function H(x,y,z) { return (x ^ y ^ z); }
	function I(x,y,z) { return (y ^ (x | (~z))); }
	function FF(a,b,c,d,x,s,ac) {
		a = AddUnsigned(a, AddUnsigned(AddUnsigned(F(b, c, d), x), ac));
		return AddUnsigned(RotateLeft(a, s), b);
	}
	function GG(a,b,c,d,x,s,ac) {
		a = AddUnsigned(a, AddUnsigned(AddUnsigned(G(b, c, d), x), ac));
		return AddUnsigned(RotateLeft(a, s), b);
	}
	function HH(a,b,c,d,x,s,ac) {
		a = AddUnsigned(a, AddUnsigned(AddUnsigned(H(b, c, d), x), ac));
		return AddUnsigned(RotateLeft(a, s), b);
	}
	function II(a,b,c,d,x,s,ac) {
		a = AddUnsigned(a, AddUnsigned(AddUnsigned(I(b, c, d), x), ac));
		return AddUnsigned(RotateLeft(a, s), b);
	}
	function ConvertToWordArray(sMessage) {
		var lWordCount;
		var lMessageLength = sMessage.length;
		var lNumberOfWords_temp1=lMessageLength + 8;
		var lNumberOfWords_temp2=(lNumberOfWords_temp1-(lNumberOfWords_temp1 % 64))/64;
		var lNumberOfWords = (lNumberOfWords_temp2+1)*16;
		var lWordArray=Array(lNumberOfWords-1);
		var lBytePosition = 0;
		var lByteCount = 0;
		while ( lByteCount<lMessageLength ) {
			lWordCount = (lByteCount-(lByteCount % 4))/4;
			lBytePosition = (lByteCount % 4)*8;
			lWordArray[lWordCount] = (lWordArray[lWordCount] | (sMessage.charCodeAt(lByteCount)<<lBytePosition));
			lByteCount++;
		}
		lWordCount = (lByteCount-(lByteCount % 4))/4;
		lBytePosition = (lByteCount % 4)*8;
		lWordArray[lWordCount] = lWordArray[lWordCount] | (0x80<<lBytePosition);
		lWordArray[lNumberOfWords-2] = lMessageLength<<3;
```

```
        lWordArray[lNumberOfWords-1] = lMessageLength>>>29;
        return lWordArray;
}
function WordToHex(lValue) {
        var WordToHexValue="",WordToHexValue_temp="",lByte,lCount;
        for (lCount = 0;lCount<=3;lCount++) {
                lByte = (lValue>>>(lCount*8)) & 255;
                WordToHexValue_temp = "0" + lByte.toString(16);
                WordToHexValue = WordToHexValue + WordToHexValue_temp.substr(WordToHexValue_temp.length-2,2);
        }
        return WordToHexValue;
}
var x=Array();
var k,AA,BB,CC,DD,a,b,c,d
var S11=7, S12=12, S13=17, S14=22;
var S21=5, S22=9 , S23=14, S24=20;
var S31=4, S32=11, S33=16, S34=23;
var S41=6, S42=10, S43=15, S44=21;
// Steps 1 and 2. Append padding bits and length and convert to words
x = ConvertToWordArray(sMessage);
// Step 3. Initialise
a = 0x67452301; b = 0xEFCDAB89; c = 0x98BADCFE; d = 0x10325476;
// Step 4. Process the message in 16-word blocks
for (k=0;k<x.length;k+=16) {
        AA=a; BB=b; CC=c; DD=d;
        a=FF(a,b,c,d,x[k+0], S11,0xD76AA478);
        d=FF(d,a,b,c,x[k+1], S12,0xE8C7B756);
        c=FF(c,d,a,b,x[k+2], S13,0x242070DB);
        b=FF(b,c,d,a,x[k+3], S14,0xC1BDCEEE);
        a=FF(a,b,c,d,x[k+4], S11,0xF57C0FAF);
        d=FF(d,a,b,c,x[k+5], S12,0x4787C62A);
        c=FF(c,d,a,b,x[k+6], S13,0xA8304613);
        b=FF(b,c,d,a,x[k+7], S14,0xFD469501);
        a=FF(a,b,c,d,x[k+8], S11,0x698098D8);
        d=FF(d,a,b,c,x[k+9], S12,0x8B44F7AF);
        c=FF(c,d,a,b,x[k+10],S13,0xFFFF5BB1);
        b=FF(b,c,d,a,x[k+11],S14,0x895CD7BE);
        a=FF(a,b,c,d,x[k+12],S11,0x6B901122);
        d=FF(d,a,b,c,x[k+13],S12,0xFD987193);
        c=FF(c,d,a,b,x[k+14],S13,0xA679438E);
        b=FF(b,c,d,a,x[k+15],S14,0x49B40821);
        a=GG(a,b,c,d,x[k+1], S21,0xF61E2562);
        d=GG(d,a,b,c,x[k+6], S22,0xC040B340);
        c=GG(c,d,a,b,x[k+11],S23,0x265E5A51);
        b=GG(b,c,d,a,x[k+0], S24,0xE9B6C7AA);
        a=GG(a,b,c,d,x[k+5], S21,0xD62F105D);
        d=GG(d,a,b,c,x[k+10],S22,0x2441453);
        c=GG(c,d,a,b,x[k+15],S23,0xD8A1E681);
        b=GG(b,c,d,a,x[k+4], S24,0xE7D3FBC8);
        a=GG(a,b,c,d,x[k+9], S21,0x21E1CDE6);
        d=GG(d,a,b,c,x[k+14],S22,0xC33707D6);
        c=GG(c,d,a,b,x[k+3], S23,0xF4D50D87);
        b=GG(b,c,d,a,x[k+8], S24,0x455A14ED);
        a=GG(a,b,c,d,x[k+13],S21,0xA9E3E905);
        d=GG(d,a,b,c,x[k+2], S22,0xFCEFA3F8);
        c=GG(c,d,a,b,x[k+7], S23,0x676F02D9);
        b=GG(b,c,d,a,x[k+12],S24,0x8D2A4C8A);
        a=HH(a,b,c,d,x[k+5], S31,0xFFFA3942);
        d=HH(d,a,b,c,x[k+8], S32,0x8771F681);
        c=HH(c,d,a,b,x[k+11],S33,0x6D9D6122);
        b=HH(b,c,d,a,x[k+14],S34,0xFDE5380C);
        a=HH(a,b,c,d,x[k+1], S31,0xA4BEEA44);
        d=HH(d,a,b,c,x[k+4], S32,0x4BDECFA9);
        c=HH(c,d,a,b,x[k+7], S33,0xF6BB4B60);
        b=HH(b,c,d,a,x[k+10],S34,0xBEBFBC70);
        a=HH(a,b,c,d,x[k+13],S31,0x289B7EC6);
        d=HH(d,a,b,c,x[k+0], S32,0xEAA127FA);
        c=HH(c,d,a,b,x[k+3], S33,0xD4EF3085);
        b=HH(b,c,d,a,x[k+6], S34,0x4881D05);
        a=HH(a,b,c,d,x[k+9], S31,0xD9D4D039);
        d=HH(d,a,b,c,x[k+12],S32,0xE6DB99E5);
        c=HH(c,d,a,b,x[k+15],S33,0x1FA27CF8);
        b=HH(b,c,d,a,x[k+2], S34,0xC4AC5665);
        a=II(a,b,c,d,x[k+0], S41,0xF4292244);
        d=II(d,a,b,c,x[k+7], S42,0x432AFF97);
        c=II(c,d,a,b,x[k+14],S43,0xAB9423A7);
        b=II(b,c,d,a,x[k+5], S44,0xFC93A039);
        a=II(a,b,c,d,x[k+12],S41,0x655B59C3);
        d=II(d,a,b,c,x[k+3], S42,0x8F0CCC92);
        c=II(c,d,a,b,x[k+10],S43,0xFFEFF47D);
        b=II(b,c,d,a,x[k+1], S44,0x85845DD1);
        a=II(a,b,c,d,x[k+8], S41,0x6FA87E4F);
        d=II(d,a,b,c,x[k+15],S42,0xFE2CE6E0);
        c=II(c,d,a,b,x[k+6], S43,0xA3014314);
```

```
            b=II(b,c,d,a,x[k+13],S44,0x4E0811A1);
            a=II(a,b,c,d,x[k+4], S41,0xF7537E82);
            d=II(d,a,b,c,x[k+11],S42,0xBD3AF235);
            c=II(c,d,a,b,x[k+2], S43,0x2AD7D2BB);
            b=II(b,c,d,a,x[k+9], S44,0xEB86D391);
            a=AddUnsigned(a,AA); b=AddUnsigned(b,BB); c=AddUnsigned(c,CC); d=AddUnsigned(d,DD);
        }
        // Step 5. Output the 128 bit digest
        var temp= WordToHex(a)+WordToHex(b)+WordToHex(c)+WordToHex(d);
        return temp.toLowerCase();
}
```