

概述

易米电话工具条api接口文件把易米工具条文件剥离了UI相关部分代码，方便用户直接通过程序调用易米工具条服务。

使用介绍

我们提供两种方式使用api接口，并提供相应示例代码。

1. 传统方式，在HTML页面中通过 `script` 标签引入api接口文件

```
<script src="https://unpkg.com/jssip-emicnet/dist/phonebar.js"></script>
<script>
  //调用 api 接口，可参见示例
</script>
```

但由于工具条api代码较复杂，我们建议用下面这种方式：

2. npm 安装易米工具条，通过各种打包编译工具使用, webpack 或gulp使用

```
1. //安装易米工具条包
npm i jssip-emicnet
2. 代码中引入
let phonebar = require("jssip-emicnet/dist/phonebar");
//或者
import phonebar from 'jssip-emicnet/dist/phonebar';
3. 注意 babel 设置，比如 target, 比如"last 5 Chrome versions",
```

注：api调用时候内部数据交互依赖浏览器的 `localStorage`。接口调用成功的用户信息都会存在 `localStorage.userData`，所以代码必须在浏览器中运行，不能在nodejs环境运行。

API 接口说明

主要接口方法说明如下：

getUser(un, pwd, switchNumber,callback) 获取用户信息接口

注：该接口会获取登录所需的所有参数 注册登陆的第一步

请求参数说明

参数	是否必须	数据类型	参数说明
un	是	String	分机号
pwd	是	String	密码
switchnumber	是	String	企业总机号
callback	是	function(err,data) {}	请求成功 err 为 null ,data如下

返回数据 data 为 JSON 格式：

属性	属性说明
inGroups	[{id: "2033", eid: "65656", name: "售前咨询"},{...}]
userData	{displayname: "****", eid: "65656", number: "1023", ...}

err 为 JSON 格式， {code: xxx,info: 'xxxxxxxxxxx'}

code	Info
50000	服务器错误
50001	参数错误
50002	获取运维地址失败
50003	分机号码或密码错误
50004	未找到该企业
50005	登录信息获取失败
其他	未知错误

调用示例

```
// getLoginData(un, pwd, switchNumber,callback)
getUser('1**6', '1**6', '0256****734',function(err,data) {
    if(err) throw new Error('请求错误')
    // 可以从 data 中获取一些可用数据, 比如:  eid,gid
})
```

init(params,eventCallback) 注册易米电话服务

注：

1. 调用 getUser 成功获取坐席信息(localStorage.userData 存入数据)就可以通过init初始化工具条。
2. init 注册登录易米服务器
3. 回调监听消息 （提供相应的回调接口才能正确使用易米电话服务）
 - 登录回调
 - 呼叫回调(呼出、呼入)
 - 被踢下线回调

参数	是否必须	数据类型	参数说明
params	是	Object	{ switchnumber, un , pwd ...}
eventCallback	是	Object	{ register : function(){...},callEvent: function() {...} }

params

参数	是否必须	数据类型	参数说明
un	是	String	分机号
pwd	是	String	密码
switchnumber	是	String	总机号
gid	是	number	当前登陆技能组 id，如果不在任何技能组填0
remoteAudio	可选	string	当前播放语音的 audio 标签 id，没传使用缺省创建的
socketUri	可选	url	网关地址，没传就用缺省的webRTC网关地址 wss://ym.vsbcc.com:9060

eventCallback

参数	是否必须	数据类型	参数说明
register	是	function	注册回调 { code: 200, data: data } code 非 200 失败
callEvent	是	function	呼入会话监听
kickedOffLine	是	function	被踢 等下线消息（比如账号过期或者其他错误）监听 data
statusChanged	否	function	如果需要监听坐席状态可以设置这个回调

回调通知说明

callEvent (type,data)

type	说明
callinResponse	座席外呼响应(内线)
calloutResponse	座席外呼响应(外线)
callinFaieldResponse	座席外呼响应(内线) 被叫超时未接听
newPBXCall	来电呼入
cancelPBXCall	坐席未接听,直接取消来电(会流转 to 空闲坐席) / 坐席未接听, 客户挂断
answeredPBXCall	通话建立
endPBXCall	通话结束

data 为 JSON格式

data	说明
a	sip 消息状态码
c	ccNumber
callTime	通话时长（只在 endPBXCall 时提供）
n	对方号码， calloutResponse/callinResponse/answeredPBXCall 事件中带

注：呼叫回调方法是使用api最复杂部分，可参见相应示例代码。

register(data)

data 为 JSON格式

data	说明
code	200 登录成功，非200登录失败
info	登录失败原因说明
data	sip相关消息，便于sip信令调试，登录成功通常不需要处理

注：主动登出请调用 `logout()`。登出正常情况下都是成功的；不成功通常是因为网络原因造成登出消息没送到服务器，但这种情况下坐席的实际状态通常也是下线的。所以调用logout可以不提供回调，如果提供，回调参数和上述登录回调参数一样。

kickedOffLine(data)

data.r 是被踢下线原因：

data.r	说明
895	其他用户登录您的账号
897	注册超时，请重新登录
898	账号过期/账号被删/账号修改
898	企业停止使用

statusChanged(data)

data.status 0 离线，1在线，2 忙碌

和调用logout情况一样，设置坐席状态正常情况下都是成功的，所以这个回调函数非必须。设置状态不成功通常是因为网络原因造成登出消息没送到服务器，但这种情况下通常也没法收到回调。

但在首次注册时候可以通过这个回调确认坐席状态的变化（从离线变成在线）。因为服务器会在注册成功后马上给客户端发坐席在线的消息通知，正式通知坐席上线。

answerPBXCall(ccNumber) 来电接听

注：

1. callEvent 回调中 newPBXCall 可以拿到 ccnumber
2. 可在 callEvent 事件中监听通话状态

调用示例

```
answerPBXCall('1564717108763414conf_1564717149723')
```

transferPBXCall(gid, transNumber, ccNumber, callback) 转接

请求参数说明

参数	是否必须	数据类型	参数说明
gid	是	string	技能组 id
transNumber	是	string	分机号
ccNumner	是	string	当前通话唯一标识
callback	否	funtion(type,data) {...}	转接成功失败回调通知

调用实例

```
transferPBXCall(2069,1023,'1564717108763414conf_1564717149723',function(res){
    if(res.type=='transferCallSuccess'){
        //成功 res.data
    }
    if(res.type=='transferCallFaild'){
        //失败 res.data
    }
})
```

call(number) 外呼

注：调用call发起呼叫后，相应的呼叫后续处理都在callEvent回调事件中。

请求参数说明

参数	是否必须	数据类型
number	是	String

返回值 2或3， 2表示呼叫外线（不用+9）， 3表示呼叫内线。调用实例

```
let calltype = phonebar.call('10010')
if (calltype ==2) console.log('呼叫外线')
```

呼叫不成功主要有以下原因（在 `calloutResponse/callinResponse`）回调里处理，参见测试用例。

```
{  
  
  '503': '对方忙碌',  
  '507': '总机号已停机',  
  '508': '非工作时间',  
  '512': '该客户今日被呼叫次数已达上限',  
  '1000': '禁止拨打无权限坐席'  
}
```

terminate(ccNumber) 挂断

注：可在 `callEvent` 事件中监听通话状态

调用示例

```
terminate('1564717108763414conf_1564717149723')
```

hold(ccNumber) 呼叫保持

unhold(ccNumber) 呼叫保持取消

logout(callback) 注销/退出

changeStaus(status) 修改坐席状态

调用示例

```
changeStaus(1) // 1 空闲 2忙碌
```

checkLogin 获取登陆状态

```
var status = checkLogin() // 0 离线 1 在线
```

注，登录成功后马上调用这个方法返回0，因为服务器会在登录后才发出在线消息，参见示例。

checkSeatState 获取坐席状态

调用示例

```
var status = checkSeatState() // 0 离线 1 空闲 2 忙碌(手动置忙, 非通话中) 3 振铃 4 通话  
5 保持
```

基本调用实现外呼

```
phonebar.getUser("1002", "1002", '02566699734', function(err,data) {  
    if (err) return  
    var groups = data.inGroups //所属技能组  
    var gid = groups && groups.length > 0 ? groups[0].id : 0  
    //注册sip  
    phonebar.init({  
        switchnumber: "02566699734",  
        gid: gid//登陆技能组  
    }, {  
        //注册回调  
        register: function (data) {  
            console.log("register", { data })  
            if (data.code == 200) {  
                //外呼  
                phonebar.call({  
                    peerID: '9183*****136',  
                    callType: 2  
                })  
            }  
        },  
        //呼入通话回调  
        callEvent: function (type, data) {  
            console.log("callEvent", { type, data })  
            if(type == 'newPBXCall') { // 来电呼入  
                var ccNumber = data.c  
                // 接听来电  
                phonebar.answerPBXCall(ccNumber)  
            }else if() {  
                //...  
            }  
        },  
    },  
}
```



```

//被踢下线消息
kickedOffLine: function (type, data) {
    console.log("kickedOffLine", { type, data })
}

})
})

```

webApiHandler(functionName,webParam) 获取技能组/坐席 (转接的时候需要)

注:

1. 当 getUser 调用成功后自动写入 `localStorage.userData` 数据, webApiHandler有些请求需要字段会从 `localStorage` 中获取。所以调用这两个方法之前要确保getUser 调用成功。
2. async/Promise 函数

请求参数说明

参数	是否必须	数据类型	参数说明
functionName	是	String	getGroups(获取所有技能组) , searchEpMembers(获取技能组包含坐席)
webParam	是	Object	技能组 {un: '分机号',pwd: pwd:"密码",eid:企业 eid} 坐席(分页获取){un,pwd,eid,searchGid,length: 每页获取个数}

调用示例

```

webApiHandler("getGroups",{
    un: 分机号,
    pwd: 密码,
    eid: eid,
}).then(res=>{
    if(res.status==200){ }
    else { } //res.status = 50001 表示 localStorage.userData没有数据
})

let result = await webApiHandler ("searchEpMembers",{
    un: 分机号,
    pwd: 密码,
    eid: eid,
    searchGid: xxx技能组id,
    length: 10// 非必填
})

```

```
if (res.status == 200) {  
  let number = res.returnData.recordsTotal //该组坐席总数  
  for (const member of res.returnData.data) { //res.returnData.data是坐席数组  
    //kefuStatus 0 离线 1 空闲 2 忙碌  
    phonebar.log(`${member.displayName} 状态 ${member.kefuStatus}`)  
  }  
}
```