概述

易米WebRTC SDK 接口 为用户提供易米**WebRTC**产品服务，让用户通过浏览器进行**点对点音频通信**。关于WebRTC和易米WebRTC产品介绍可参见这份文档 ，本文档描述sdk提供的各个api接口以及通话呼叫的流程。

使用方式

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>
   ```

   但要注意易米有两套呼叫平台系统，对应不同客户和使用户场景。SDK 1.x版本使用 呼叫平台 1；SDK 2.x版本使用 呼叫平台 2

   CDN服务器https://unpkg.com 会重定向当前发布最新版本，可能是 1.x 也可能是 2.x。 所以如果明确使用某个版本，比如 1.5.4 （1.x 当前最新版本） 或者 2.1.0（当前2.x最新版本） 请直接写明使用版本 `<script src='https://unpkg.com/jssip-emicnet@1.5.4/dist/phonebar.js'></script>`

   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：

1. 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。
2. WebRTC **必须使用https**，本地开发使用 `localhost` 可以跳过https限制，但是需要手动设置允许访问麦克；实际部署必须使用https，并允许访问麦克风设置（缺省是不允许访问麦克风）

API 说明

变更说明

易米有两套呼叫平台系统，对应不同客户和使用户场景。API 1.x版本使用 呼叫平台 1；API 2.x版本都使用 呼叫平台 2。我们会尽量做到1.x 版本和 2.x版本接口一致。

api 2.x 版本相对于 1.x 版本的变更如下：

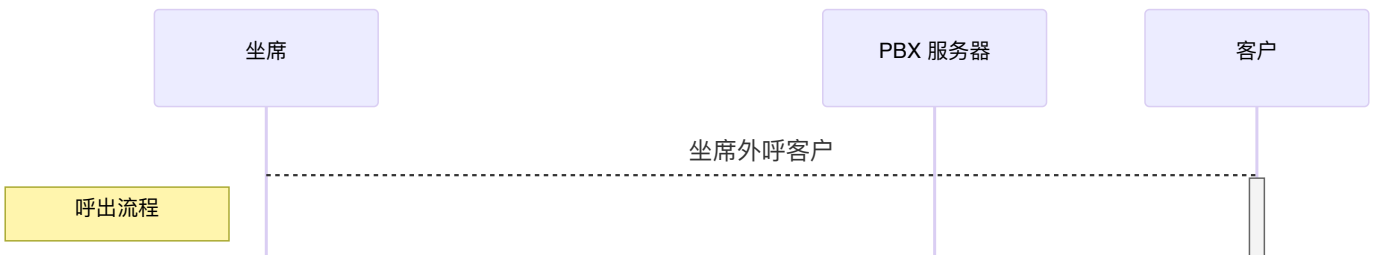
1. `getUser2` 接口输入增加 `backend` 参数 , 设定 api接口向哪个服务器发起 http请求。如果没有设置（包括使用 `getUser` 接口）则使用易米缺省的生产服务器平台。
2. `getUser` 成功调用获取客户信息后会存入 localstorage.userData, userData返回字段和1.x 基本相同，只有少数几个字段稍微做了改变：比如技能组信息叫`groupInfo` 。
3. `init` 接口 1.x 版本的需要输入参数 socketUri 网关地址，在2.x版本不需要这个参数。
4. 1.x 主要为转接提供的查询技能组和坐席信息的接口 `webApiHandler("getGroups", {})` 和 `webApiHandler ("searchEpMembers",{ } )` 改为 `getGroups` 和 `getGroupMembers`

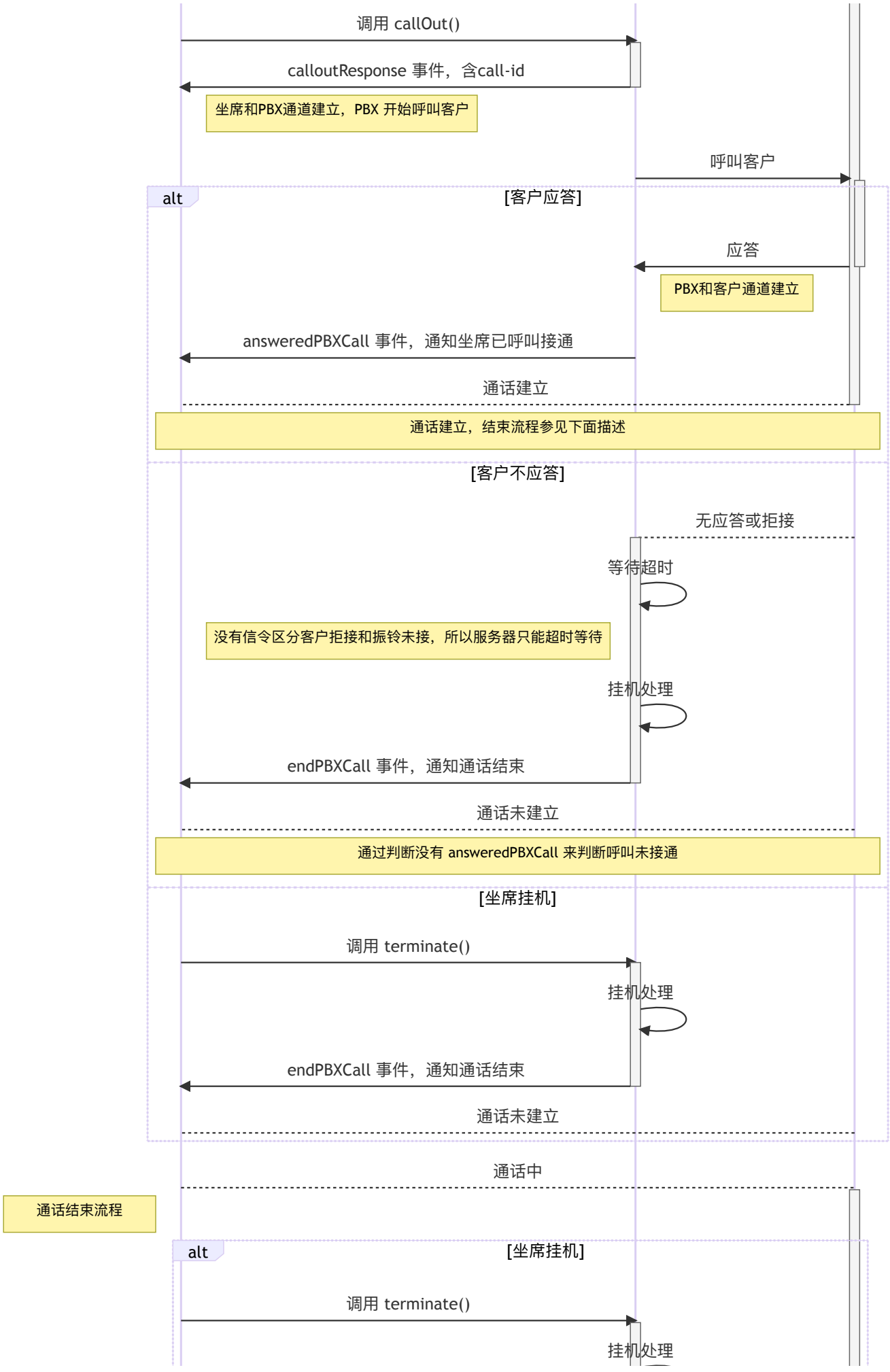
呼叫流程图

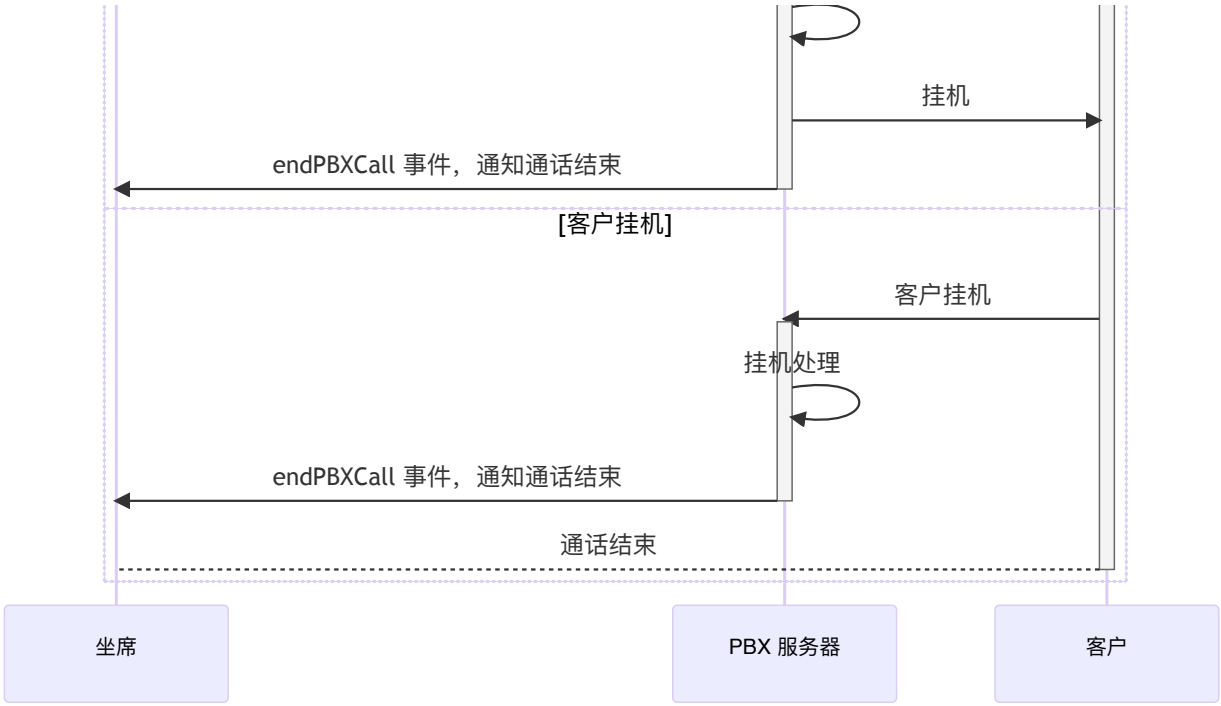
对呼叫流程了解是正确使用api接口的关键。呼叫流程分**呼出**和**呼入**流程，在通话建立前分别有不同的处理分支；当通话建立后，它们结束通话流程是一致。整体上WebRTC的呼叫流程和我们使用手机拨打、接听电话的流程并没有太大差异，对照我们平时打电话过程可以加深对通话流程的认知。

呼出

注：alt 表示几个不同处理分支



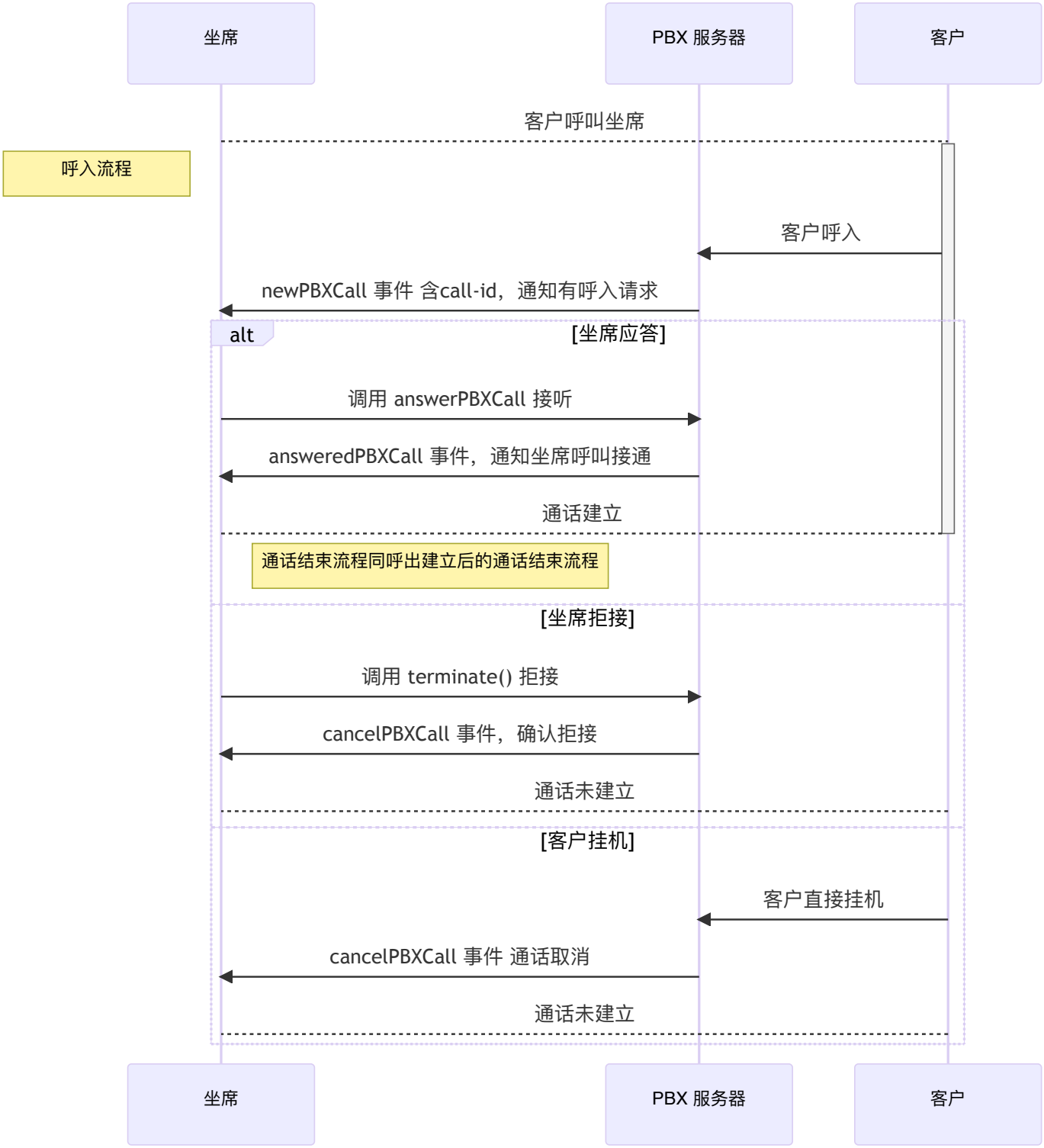

调用 callOut()
calloutResponse 事件，含call-id
坐席和PBX通道建立，PBX 开始呼叫客户
呼叫客户
alt
[客户应答]
应答
PBX和客户通道建立
answeredPBXCall 事件，通知坐席已呼叫接通
通话建立
通话建立，结束流程参见下面描述
[客户不应答]
无应答或拒接
等待超时
没有信令区分客户拒接和振铃未接，所以服务器只能超时等待
挂机处理
endPBXCall 事件，通知通话结束
通话未建立
通过判断没有 answeredPBXCall 来判断呼叫未接通
[坐席挂机]
调用 terminate()
挂机处理
endPBXCall 事件，通知通话结束
通话未建立
通话中
通话结束流程
alt
[坐席挂机]
调用 terminate()
挂机处理



注：

1. 呼出流程和我们普通接电话有一点不同：对于voip呼叫运营商没有信令区分客户**拒接**和振铃中**未接**，所以我们就没法区分这两种情况。表现就是主叫会一直听到振铃声，即便被叫已经选择拒接，直到被叫**接起**或者振铃**超时**。振铃超时我司才能判断呼叫未建立，通过 `endPBXCall` 事件通知通话结束。
2. 通话结束后 2.x 版本会收到设置坐席空闲的消息，但是 1.x 版本不会有这条消息。

呼入


坐席
PBX 服务器
客户
客户呼叫坐席
呼入流程
客户呼入
newPBXCall 事件 含call-id，通知有呼入请求
alt
[坐席应答]
调用 answerPBXCall 接听
answeredPBXCall 事件，通知坐席呼叫接通
通话建立
通话结束流程同呼出建立后的通话结束流程
[坐席拒接]
调用 terminate() 拒接
cancelPBXCall 事件，确认拒接
通话未建立
[客户挂机]
客户直接挂机
cancelPBXCall 事件 通话取消
通话未建立
坐席
PBX 服务器
客户

注：呼入流程和我们普通接电话有一点不同就是，我们通常是不允许坐席拒接呼入电话（通过ivr设置），所以如果坐席调用 `terminate()` 拒接，服务器会马上再打给他/她，让坐席感觉好像没法拒接电话。

API 接口描述

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

注：该接口获取登录所需的所有参数，是使用易米 webrtc api的第一步 如果需要输入更多参数建议使用 `getUser2` 接口

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**，并存入 localstorage.userData．2.x 版本返回的 userData数据属性比1.x版本多，但api用到的属性没有太多变化，示例均有说明：

属性	属性说明
groupInfo	[{ gid:1000003232, eid: "65656", name: "售前咨询"},{...}]
userInfo	{displayname: "***", eid: "65656", number: "1023", ...}
enterprise	{extension_start，extension_end ...} 企业属性,比如分机号号段
其他属性	其他属性详情参见 Local Storage

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

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

```
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
})
```

getUser2({ un, pwd, switchnumber, callintype , number,backend }, callback) 支持sip话机和回拨话机，获取用户信息接口

请求参数说明

un,pwd, switchnumber,callback 和 getUser 的含义相同，也都是必填参数

参数	是否必须	数据类型	参数说明
callintype	否	number(int)	5 是sip话机，4回拨话机，2是原有的voip，其他输入返回错误
number	否	String	sip话机或回拨话机号码，如果callintype是4或5，number必填
backend	否	String	哪个服务器发起请求，不填则api请求发往易米缺省的生产服务器平台

从1.4.0 开始， api接口支持sip话机和回拨话机功能。使用它们，代码和之前**唯一不同**就是获取用户信息接口请调用`getUser2`. 原有接口`getUser`依然正常使用，但它只支持获取voip登录的相关信息。调用`getUser2`成功后，其他操作（登录、打电话）都和原来相同，所以代码不需要做别的改动。

用户拨打电话的界面操作也和原来相同，只是总机会回拨到sip话机（相比于之前回拨到浏览器）。用户可以在界面以及sip话机上挂断电话、通话保持。另有文档描述如何配置sip话机。

`getUser2`采用es6对象解构赋值的方法传递参数，对象属性名称都是**小写**；调用传递参数如果用es6对象构建的简洁写法 Shorthand property names 注意变量名都要**小写**：

```
let un='1**6',pwd='1**6',switchnumber = '0256****734', callintype = 5, number = 'xxxx'
getUser2({un,pwd,switchnumber,callintype,number},function(err,data) {...})
```

2.x 版本增加 backend参数，设置连接的服务器。

详见 https://github.com/emicnet/test-jssip-emic 测试代码

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 设置各种回调函数。当参数输入正确，init发起注册请求，尝试登录易米服务器并返回true，注册结果通过回调函数通知。当缺少必要参数，不能发起注册请求就返回false。
3. init 接口 和 getUser 接口分开而不是合并成一个接口主要原因是当坐席处在多个技能组，需要用户输入登录使用的技能组gid，gid参数是init 的必填参数。
4. 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 auido 标签 id，没传使用缺省创建的 标签 id
callintype	可选	number	sip话机是5，回拨话机是4。voip是2或者不填，其他输入返回错误
number	可选	String	sip话机或回拨话机号码，如果callintype是4或5，number必填

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)
callinFaildResponse	座席外呼响应(内线) 被叫超时未接听
newPBXCall	来电呼入，UI程序响应这个消息提示来电
cancelPBXCall	呼入取消事件：坐席拒接(会流转到其他空闲坐席) /客户挂断，**UI程序需要响应这个消息**，更新UI界面，比如取消呼入来电提醒
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 当前通话唯一标识
callTime	通话时长，只在 endPBXCall 时提供，参见下面 r 的描述
n	对方号码， calloutResponse/callinResponse/answeredPBXCall 事件中带
info	呼叫失败原因说明；呼叫成功没有info字段
r	数字字符串，在 endPBXCall 时提供，表示呼叫结束原因, "25"表示客户挂断，"26"表示坐席挂断，"20"表示呼叫未接通。只有r为 25 和 26 时候通话才真正建立，callTime才表示实际通话时长

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。`endPBXCall`回调事件要判断 `data.r`结束原因，只有 25或26 callTime才表示实际通话时长，其他情况 callTime的收到 `endPBXCall`事件的时间戳。

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，成功没有info字段
cause	失败原因码
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。

threewayCall({ccNumber,result})

三方通话结果回调，回调输入参数采用es6对象解构赋值，参见 **三方呼叫** 章节

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')
```

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。参见调用实例

如果number 非数字直接返回

```
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

threewayCall({ccNumber,type,callee}) 三方呼叫

我们支持三种形式的三方通话功能：1. 转到某个特定坐席，即邀请另一个坐席一起参与通话；2. 一种转到某个技能组，看看这个技能组谁能接起；3. 转外线，就是请外部的人来参与通话。 type 1, 2,3 分别表示这三种情况。type 4 是挂断三方，参见下面描述。

请求参数说明，参数采用es6对象解构赋值的方法传递

参数	是否必须	数据类型	参数说明
ccNumber	是	string	当前通话唯一标识；如果成功邀请三方加入，同样使用该ccNumber作为三方通话唯一标识
type	是	number	三种形式三方通话：坐席 1 、技能组 2 、外线 3； 4 是挂断三方
callee	是	string	type 1时为分机号；type 2 时为技能组 id；typ3 时为外线号码

`threewayCall({ccNumber,type,callee})` 是异步函数，需要`await` 它的处理结果，返回结果是 `{result,reason}` 的object

result == 0 表示调用成功，服务器处理了这个三方呼叫的请求，并向第三方发起呼叫。对三方呼叫的实际结果通过三方回调方法告知。

result ==1 表示三方处理失败，reason是失败原因。常见原因有：转坐席但该坐席处在非空闲转态；转技能组，但该技能组id不正确（比如不存在），转外线但外线号码有误。

调用示例

```
  let {result,reason} = await
phonebar.threewayCall({ccNumber,type:3,callee:threeWayNumber})
  if (result == 0) {
      // 已处理这个三方呼叫请求，后续处理结果通过 threewayCallResult 回调告知
  } else {
      // 我方服务器未能发起三方呼叫请求，原图：${reaosn}
  }

  //三方呼叫结果回调函数
```

```
function({ccNumber,result}) {
    //回调函数可校验 ccNumber
    if (result == 0) {
       //三方呼叫成功，三方已进入电话会议
    }
    if (result == 1) {
       //三方呼叫失败
    }
    if (result == 2) {
       //三方主动退出，即三方挂机，通话还原为原来两方通话形式
    }
}
```

另需要注意的，我方服务器呼叫三方外线号码和正常呼叫外线号码流程是一样的，所以我们们没法区分用户**拒接**和振铃中**未接**这两种情况。表现就是已在通话中双方会一直听到振铃声（即便三方已经选择拒接来电）直到三方**接起**或者振铃**超时** （参见前面章节的呼出流程图）。

三方呼叫的实际结果通过三方回调通知

取消/结束三方通话

`threewayCall({ccNumber,type:4})` 取消/结束三方，传两个参数 ccNumber 和 type: 4 。如果三方已经在通话中，就把第三方移出通话，通话还原成原来的两方通话；如果三方还未进入通话，即意味着服务器还在呼叫三方，就取消对三方的呼叫，通话同样还原为原来的两方通话。

另外需要注意，三方通话中如果主叫调用 `terminate(ccNumber)` 只是结束和原来被叫的通话，但被叫和三方通话还在，所以直接调用 `terminate(ccNumber)` 三方通话就会变成**原被叫和第三方的两方通话**，坐席没法再参与或者结束该通话。

正确结束三方通话的步骤是先调用 `threewayCall({ccNumber,type:4})` 把三方变回原来两方，然后再调用 `terminate(ccNumber)` 结束两方通话。 完整例子详见 https://github.com/emicnet/test-jssip-emic 示例代码(newcc分支)。 **没有直接结束三方通话的接口调用。**

注：三方通话是相对复杂的通话流程，可以参照iPhone 三方通话流程加深理解。但注意安卓手机的三方通话成功率略低，容易出现“无法要求xxx进入电话会议提示”；非同一家运营商的号码也容易失败，出现同样的提示。

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
tranNumber	是	string	分机号
ccNumber	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

调用示例

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

logout(callback) 注销/退出

注，2.x版本web 调用必须有合法token，每个token有一定有效期，所以在init登录成功后SDK会起定时器定时刷新token， 坐席下线一定要记得调用 `logout` SDK才会清除定时器。

changeStaus(status) 修改坐席状态

调用示例，status值非数字直接返回

```
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

完整例子详见 https://github.com/emicnet/test-jssip-emic 示例代码

```
phonebar.getUser("1002", "1002", '02566699734', function(err,data) {
    if (err) return
    var groups = data.groupInfo //所属技能组
    //简单选取坐席所在的第一个技能组，更复杂示例参见 test-jssip-emic 测试代码
    var gid = groups && groups.length > 0 ? groups[0].id : 0
    //注册sip
    phonebar.init({
      un:1006
      pwd:12345678
      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 })
      }

    })
})
```

getGroups(page) 获取企业技能组列表

1. getGroups 必须在当 getUser 调用成功后自动写入 `localstorage.userData` 数据才能调用
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
page	否	int	获取第几页组信息，不填就获取第一页信息

返回值是JSON 数据，具体使用参见示例，其中最重要的属性有：

```
{  current_page:xxx, //current_page 为传入参数，缺省是 1 ，即第一页数据，注意 下标从1开始，而不是从0开始
   last_page:xxx, //last_page 表示一共几页,如果 last_page 比 current_page 大 就可以再次调用获取
   total:xxx //技能组总数
   data:[]//技能组数组,数组元素信息如下
   { ccgeid: 243,
     gid: 1000001728,
     group_real_time_state: {... state_id: 1, state_name: "有空闲"},
     id: 762,
     name: "G0515B",
     seid: 134}
}
```

getGroupMembers({gid,seat_page}) 获取技能组成员

请求参数说明

参数	是否必须	数据类型	参数说明
gid	是	int	技能组的gid
seat_page	否	int	获取第几页组信息，不填就获取第一页信息

获取gid技能组里的成员信息，返回值 res.data 是数组，针对输入参数gid的情况，返回具体数据为 data[0]

具体使用参见示例

```
data: [{id: 1151, gid: 1000003650, name: "10804-test", seats: [,…]}]
0: {id: 1151, gid: 1000003650, name: "10804-test", seats: [,…]}
gid: 1000003650,
id: 1151,
name: "10804-test",
current_page: //为传入参数，缺省是 1 ，即第一页数据，注意 下标从1开始，而不是从0开始
last_page:xxx, //last_page 表示一共几页,如果 last_page 比 current_page 大 就可以再次调用获取
total:xxx //改技能组坐席总数
seats: [,…] 该技能组的坐席信息
{
    displayname: "6666",
    id: 1000962235,
    number: "6666",
    seat_real_time_state: {uid: 1000962235, state_id: 1, state_name: "离线"},
    seid: 134,
    work_number: "ldd"
}
```

注：getGroups 和 getGroupMembers 主要是在转接时候用到, 和1.x 版本有不同。